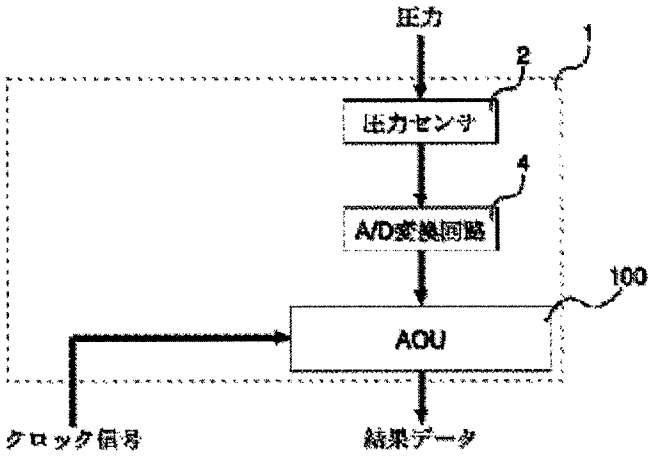
TOUCH SENSOR

Publication number: JP2001242023
Publication date: 2001-09-07
Inventor: AJIOKA YOSHIAKI
Applicant: ECCHANDESU KK
Classification:
- international: ***G01L5/00; G01B7/00; G01B7/28; G01B21/00; G01B21/20;*** **G01L5/00; G01B7/00; G01B7/28; G01B21/00; G01B21/20;** (IPC1-7): G01L5/00; G01B7/00; G01B7/28; G01B21/00; G01B21/20
- European:
Application number: JP20000052735 20000229
Priority number(s): JP20000052735 20000229

Report a data error here

Abstract of **JP2001242023**

PROBLEM TO BE SOLVED: To design and manufacture a touch sensor having a pressure sensor, an A/D converting circuit, and an arrangement operation unit arranged into a two-dimensional grid. SOLUTION: A function column 1 is provided with the pressure sensor 2, the A/D converting circuit 4, and the arrangement operation unit 100 as shown in Figure 1. The function column 1 is arranged into the two-dimensional grid as shown in Figure 5 and a signal wire is laid between function columns 1 in 4 circumferences. When the respective function columns 1 are provided with oscillation circuits 5 as shown in Figure 2, the signal wire for feeding the same clock signals to all the function columns 1 from the outside can be eliminated. The arrangement operation unit 10 outputs pixels to be outputted, in parallel.



Data supplied from the ***esp@cenet*** database - Worldwide

(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号
特開2001－242023
（P2001－242023A）
(43)公開日　平成13年９月７日(2001.9.7)

(51)Int.Cl.'		識別記号	FI		テーマコード'(参考)	
Ｇ０１Ｌ	5/00	１０１	Ｃ０１Ｌ	5/00	１０１Ｚ	２Ｆ０５１
Ｇ０１Ｂ	7/00		Ｃ０１Ｂ	7/00	Ｎ	２Ｆ０６３
	7/28			7/28	Ｈ	２Ｆ０６９
	21/00			21/00	Ｄ	
	21/20			21/20	Ｆ	

審査請求　未請求　請求項の数５　ＯＬ　（全 37 頁）

(21)出願番号　特願2000－52735(P2000－52735)

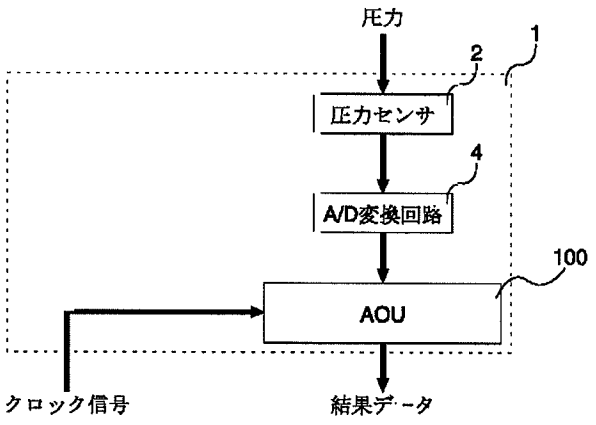
(22)出願日　平成12年２月29日(2000.2.29)

(71)出願人　39805/167
株式会社エッチャンデス
愛知県蒲郡市中央本町12番７号
(72)発明者　味岡　義明
愛知県蒲郡市中央本町12番７号

最終頁に続く

(54)【発明の名称】　タッチセンサ

(57)【要約】
【目的】　圧力センサ、Ａ／Ｄ変換回路、配列演算ユニットを二次元格子状に配列したタッチセンサを設計及び製造する。
【構成】　図１に示すように、機能コラム１は圧力センサ２、Ａ／Ｄ変換回路４、及び配列演算ユニット１００を備えている。また図５に示すように、機能コラム１は二次元格子状に配列され、４近傍の機能コラム１同士の間で信号線が配線される。図２に示すように、各々の機能コラム１が発振回路５を備えた場合、外部から全ての機能コラム１に同一のクロック信号を供給するための信号線が必要がなくなる。配列演算ユニット１００から、出力すべき画像の画素が並列に出力される。



【特許請求の範囲】
【請求項１】　１個の画素に対して画像処理を行う機能コラムが二次元格子状に配列されたタッチセンサであって、前記機能コラムが、
１～１６個の圧力センサと、
１～１６個のA／D変換回路と、
１～３２個の配列演算ユニットと、を備え、前記配列演算ユニットが局所並列画像処理を実行することを特徴とするタッチセンサ。
【請求項２】　請求項１記載の機能コラムに対して、１個の発振回路を備えたことを特徴とするタッチセンサ。
【請求項３】　請求項１及び２記載のタッチセンサに対して、
前記機能コラム内で前記圧力センサが前記A／D変換回路にアナログ信号を送信するように前記信号線が配線されることと、
前記機能コラム内で前記A／D変換回路が前記配列演算ユニットの１個にデジタル信号を送信するように前記信号線が配線されることと、
前記機能コラム内で前記発振回路が前記配列演算ユニットの全てにクロック信号を送信するように前記信号線が配線されることと、
前記配列演算ユニットが２～３２個の場合、前記機能コラム内で前記配列演算ユニットの各々が残りの前記配列演算ユニットのうち１～３１個と前記デジタル信号を通信するように前記信号線が配線されることと、
前記配列演算ユニットの各々が、二次元格子上で４近傍に位置する前記機能コラムの各々が備える１個の前記配列演算ユニットと前記デジタル信号を通信するように前記信号線が配線されることと、を特徴とするタッチセンサ。
【請求項４】　請求項１、請求項２、及び請求項３記載の機能コラムに対して、圧力センサのうち１～１６個の代りに１～１６個の温度センサを備えたことを特徴とするタッチセンサ。
【請求項５】　請求項１から請求項４記載の機能コラムに対して、圧力センサのうち１～１６個の代りに１～１６個及び１～１６対の電極を備えたことを特徴とするタッチセンサ。
【発明の詳細な説明】
【０００１】
【発明の属する技術の分野】本発明は、画像中の物体の位置及び大きさを検出して物体をパターンマッチングするタッチセンサに関し、詳しくは、圧力センサ、温度センサ及び電極を介して物体の圧力、温度、水分及び電気抵抗を画像化することにより、これらの画像から構成される動画像中の物体のエッジ情報から、物体の位置及び大きさを検出し、物体のパターンマッチングを行い、物体のマッチング結果、位置及び大きさを出力するものに関する。
【０００２】
【従来の技術】近年、圧電素子、圧電振動子、超音波振動子、導電性フィルム、光ファイバ、受光素子、及び電界効果型トランジスタ（ＭＯＳＦＥＴ）等を用いた高特性圧力センサ及び触覚センサが多数開発されるようになってきた（例えば、特開平０５－１４９８０９、特開平０５－２１５６２５、特開平０５－２８８６１９、特開平０７－０１９９７５、特開平０７－１１５２０９、特開平０９－２０３６７１、特開平１１－０４４５８７、特開平１１－０８８６０１参照）。また圧力センサ及び触覚センサを二次元格子状に配列して圧力の位置及び大きさを検出するタッチセンサも開発されている（例えば、特開平０５－１４９８０９、特開平０５－２１５６２５、特開平０６－０１１４０３、特開平０６－０１８２１６、特開平０７－０１９９７５、特開平０７－１９０８７０、特開平１１－１１８４６０、特表平１１－５０６５５９参照）。この他に、焦電素子、赤外線受光素子、光導波路、光シャッタ素子、サーミスタ及び熱電対等の温度センサを用いたもの（例えば、特開平０５－００３３４６、特開平０７－１８１０８７、特開平０８－０７６１７４、特開平０８－１２２２４６、特開平０８－３２０２５８、特開平０９－０３３２１５、特開平０９－０３５０３８、特開平１０－１８１０９８、特開平１１－３５４８２９参照）、及び電極を用いたもの（例えば、特開平０５－０１０８２６、特開平１０－２４０４３５、特開平１０－３１４１４８、特開２０００－００５１４７参照）も開発されている。しかしながら上記のようなタッチセンサでは、各種センサと、入力刺激の位置、大きさ及び動きを検出したり入力刺激に対してパターンマッチングをする装置とが独立に存在するため、各種センサと装置の間の通信がボトルネックになったり、入力刺激が二次元であることを装置が利用できなかったり、さらには配線及び付加回路のために装置自体が大きくなってしまうといった問題が生じる。
【０００３】一方で、本発明者は既に、エッジ情報生成装置（特願平１１－１４５６３８参照）、エッジ情報形成装置（特願平１１－２０９７３８参照）、物体／背景分離装置（特願平１０－２５７３２７参照）、位置／大きさ検出装置（特願平１１－２５０９９０参照）、領域正規化装置（特願平１１－２５０９８６参照）、及び視覚装置（特願平１１－２５３６３４参照）を開発している。これらの装置は統一的なアーキテクチャー上で画素単位で並列に画像処理を行い、ノイズの多い実環境で移動物体及び静止物体のエッジ情報を生成し、移動物体及び静止物体の領域を背景から分離し、移動物体及び静止物体の領域の位置及び大きさを検出し、移動物体及び静止物体の領域を画像サイズに正規化してパターンマッチングをすることができる。さらに視覚装置を電荷結合素子（ＣＣＤ）及びフォトダイオードと共に大規模集積回路（ＬＳＩ）に組み込むことにより、高機能イメージセ

ンサーができる（特願平１１－３６９２２０参照）。

【０００４】これらのことを考慮すると、ＬＳＩ実装技術を用いて、圧力センサ、温度センサ及び電極と共に視覚装置の局所並列画像処理をＬＳＩに実装することができれば、実時間で移動物体及び静止物体のマッチング結果、位置及び大きさを出力することができる高機能なタッチセンサを製造できると期待される。

【０００５】

【発明が解決しようとする課題】そこで、請求項記載の本発明は、二次元格子状に配列した圧力センサ、温度センサ及び電極等の各種センサ、Ａ／Ｄ変換回路、及び配列演算ユニットをＬＳＩに実装し、局所並列画像処理を実行することにより、移動物体及び静止物体を射影してエッジ情報を生成し、移動物体及び静止物体の領域を背景から分離し、移動物体及び静止物体の領域の位置及び大きさを検出し、移動物体及び静止物体の領域を画像サイズに正規化してパターンマッチングをすることを目的とする。

【０００６】

【課題を解決するための手段】請求項１の発明は、１個の画素に対して画像処理を行う機能コラムが二次元格子状に配列されたタッチセンサであって、前記機能コラムが、１～１６個の圧力センサと、１～１６個のＡ／Ｄ変換回路と、１～３２個の配列演算ユニットと、を備え、前記配列演算ユニットが局所並列画像処理を実行することを特徴とするタッチセンサである。本発明は、圧電素子、圧電振動子、超音波振動子、導電性フィルム、光ファイバ、受光素子、及び電界効果型トランジスタ（ＭＯＳＦＥＴ）等から構成される前記圧力センサを用いて圧力を検知することにより、前記圧力センサからアナログ信号を取り出し、前記Ａ／Ｄ変換回路を用いて前記アナログ信号をデジタル信号に変換し、前記デジタル信号を前記配列演算ユニットに入力する。前記配列演算ユニットは、画像記憶手段、画像振動手段、エッジ情報生成手段、エッジ情報形成手段、物体／背景分離手段、位置／大きさ検出手段、領域正規化手段、及びパターンマッチング手段を用いて前記局所並列画像処理を実行し、前記デジタル信号によって構成されるデジタル画像中に投影された移動物体及び静止物体の位置及び大きさを検出し、前記移動物体及び前記静止物体のパターンマッチングを行う。前記配列演算ユニットは、前記画像記憶手段、前記画像振動手段、前記エッジ情報生成手段、前記エッジ情報形成手段、前記物体／背景分離手段、前記位置／大きさ検出手段、前記領域正規化手段、及び前記パターンマッチング手段の処理結果の各々を、前記デジタル画像、振動画像、粗エッジ情報画像、形成エッジ情報画像、物体領域画像、重複情報画像、正規化画像、及びマッチング結果画像の画素として、画像保持手段を用いて出力することができる。したがって前記タッチセンサの設計及び製造に関する諸問題が好適に解決される。

【０００７】請求項２の発明は、請求項１記載の機能コラムに対して、１個の発振回路を備えたことを特徴とするタッチセンサである。本発明は、前記機能コラムの全てが独立した前記発振回路を備えることにより、大規模集積回路に対してクロックスキューなどの遅延時間問題及び配線問題を回避することができる。したがって前記タッチセンサの設計及び製造に関する諸問題が好適に解決される。

【０００８】請求項３の発明は、請求項１及び２記載のタッチセンサに対して、前記機能コラム内で前記圧力センサが前記Ａ／Ｄ変換回路にアナログ信号を送信するように前記信号線が配線されることと、前記機能コラム内で前記Ａ／Ｄ変換回路が前記配列演算ユニットの１個にデジタル信号を送信するように前記信号線が配線されることと、前記機能コラム内で前記発振回路が前記配列演算ユニットの全てにクロック信号を送信するように前記信号線が配線されることと、前記配列演算ユニットが２～３２個の場合、前記機能コラム内で前記配列演算ユニットの各々が残りの前記配列演算ユニットのうち１～３１個と前記デジタル信号を通信するように前記信号線が配線されることと、前記配列演算ユニットの各々が、二次元格子上で４近傍に位置する前記機能コラムの各々が備える１個の前記配列演算ユニットと前記デジタル信号を通信するように前記信号線が配線されることと、を特徴とするタッチセンサである。本発明は、前記信号線が前記機能コラム内と、前記二次元格子上で互いに前記４近傍に位置する前記機能コラム間で配線される。つまり前記機能コラムが互いに前記４近傍に位置しない前記機能コラム間で前記信号線が配線されないので、大規模集積回路の配線パターンは単純化され、しかも長距離配線による遅延時間を考慮する必要がなくなる。したがって前記タッチセンサの設計及び製造に関する諸問題が好適に解決される。

【０００９】請求項４の発明は、請求項１、請求項２、及び請求項３記載の機能コラムに対して、圧力センサのうち１～１６個の代りに１～１６個の温度センサを備えたことを特徴とするタッチセンサである。本発明は、焦電素子、赤外線受光素子、光導波路、光シャッタ素子、サーミスタ及び熱電対等から構成される前記温度センサを用いて温度を検知することにより、前記温度センサからアナログ信号を取り出し、前記Ａ／Ｄ変換回路を用いて前記アナログ信号をデジタル信号に変換し、前記デジタル信号を前記配列演算ユニットに入力する。圧力センサ及び温度センサが１つの機能コラムに含まれても良い。これにより前記タッチセンサの性能は向上する。前記配列演算ユニットは、画像記憶手段、画像振動手段、エッジ情報生成手段、エッジ情報形成手段、物体／背景分離手段、位置／大きさ検出手段、領域正規化手段、及びパターンマッチング手段を用いて前記局所並列画像処理を実行し、前記デジタル信号によって構成されるデジ

タル画像中に射影された移動物体及び静止物体の位置及び大きさを検出し、前記移動物体及び前記静止物体のパターンマッチングを行う。前記配列演算ユニットは、前記画像記憶手段、前記画像振動手段、前記エッジ情報生成手段、前記エッジ情報形成手段、前記物体／背景分離手段、前記位置／大きさ検出手段、前記領域正規化手段、及び前記パターンマッチング手段の処理結果の各々を、前記デジタル画像、振動画像、粗エッジ情報画像、形成エッジ情報画像、物体領域画像、重複情報画像、正規化画像、及びマッチング結果画像の画素として、画像保持手段を用いて出力することができる。したがって前記タッチセンサの設計及び製造に関する諸問題が好適に解決される。

【００１０】請求項５の発明は、請求項１から請求項４記載の機能コラムに対して、圧力センサのうち１～１６個の代りに１～１６個及び１～１６対の電極を備えたことを特徴とするタッチセンサである。本発明は、前記電極を用いて電気及び抵抗を検知することにより、前記電極からアナログ信号を取り出し、前記Ａ／Ｄ変換回路を用いて前記アナログ信号をデジタル信号に変換し、前記デジタル信号を前記配列演算ユニットに入力する。圧力センサ、温度センサ及び電極が１つの機能コラムに含まれても良い。これにより前記タッチセンサの性能は向上する。前記配列演算ユニットは、画像記憶手段、画像振動手段、エッジ情報生成手段、エッジ情報形成手段、物体／背景分離手段、位置／大きさ検出手段、領域正規化手段、及びパターンマッチング手段を用いて前記局所並列画像処理を実行し、前記デジタル信号によって構成されるデジタル画像中に射影された移動物体及び静止物体の位置及び大きさを検出し、前記移動物体及び前記静止物体のパターンマッチングを行う。前記配列演算ユニットは、前記画像記憶手段、前記画像振動手段、前記エッジ情報生成手段、前記エッジ情報形成手段、前記物体／背景分離手段、前記位置／大きさ検出手段、前記領域正規化手段、及び前記パターンマッチング手段の処理結果の各々を、前記デジタル画像、振動画像、粗エッジ情報画像、形成エッジ情報画像、物体領域画像、重複情報画像、正規化画像、及びマッチング結果画像の画素として、画像保持手段を用いて出力することができる。したがって前記タッチセンサの設計及び製造に関する諸問題が好適に解決される。

【００１１】

【発明の実施の形態】以下、本発明のタッチセンサの実施形態を挙げ、図面を参照して説明する。

【００１２】まず請求項１及び２記載の発明は各々図１及び２に示すような機能コラム１を二次元格子状に配列することによって実現される。なお図１及び２において、配列演算ユニット１００をＡＯＵと略記する。圧力センサ２及びＡ／Ｄ変換回路４の対の数が２個以上の場合、この対は図１及び２の圧力センサ２及びＡ／Ｄ変換回路４の対に対して並列に配置される。つまり圧力センサ２及びＡ／Ｄ変換回路４の対は実装面に対して二次元に配列される。配列演算ユニット１００の数が２個以上の場合、図１及び２の配列演算ユニット１００を根とする木構造のように配置される。具体的には、機能コラム１は図３及び４に示すようになる。図３に示した機能コラム１はタッチセンサに触れた物体の位置及び接触面積を検知するためのものであり、４に示した機能コラム１はタッチセンサに触れた移動物体の位置及び接触面積を検知するためのものである。なお、図１及び２に示された１個の配列演算ユニット１００でも、プログラムによって複数の手段を実現できる。

【００１３】請求項３記載の発明では、図１から４までに示すような機能コラム１を二次元格子状に配列した場合、図５に示すように信号線が配線される。なお図５において、機能コラム１をＦＣと略記する。また電源線及びクロック信号は省略されている。図５に示された信号線は実際には配列演算ユニット１００に接続されている。図５から明らかなように、機能コラム１間の信号線は、他の機能コラム１間の信号線と交差しない。しかも配列演算ユニット１００は同じ画像処理手段を実現する配列演算ユニット１００とだけ通信をするので、たとえ機能コラム１に複数の配列演算ユニット１００が含まれるとしても、これらの信号線は機能コラム１間では交差しない。図６に示すように、機能コラム１中の全ての配列演算ユニット１００が三次元大規模集積回路（三次元ＶＬＳＩ）によって垂直に並ぶように実装されれば、機能コラム１間の信号線は配列演算ユニット１００間ですら交差しない。ただし図６では、同じ高さに位置する配列演算ユニット１００は同じ画像処理手段を実現するものとする。したがってイメージセンサの設計及び実装は極めて容易になる。

【００１４】請求項４及び５記載の発明は、請求項１、２、及び３記載の機能コラム１において、圧力センサ２の代りに温度センサ及び電極を実装したものである。したがって配列演算ユニット１００及び配線に関して特に変更はない。ただし相補性金属酸化膜半導体（ＣＭＯＳ）によって実現される温度センサ及び電極を用いた場合、設計工程及び製造工程が簡単になる。

【００１５】ここまでは、請求項１から５記載のタッチセンサに対して、二次元格子状に配列された機能コラム１の構成と、配列演算ユニット１００間の信号線の配線について説明してきた。配列演算ユニット１００は、これらの信号線を介して４近傍の配列演算ユニット１００と画像データを通信することにより、以下のような画像処理手段を実行することができる。

【００１６】配列演算ユニット１００が実行する画像記憶手段１２、画像振動手段、エッジ情報生成手段１４、エッジ情報形成手段１５、物体／背景分離手段１６、領域正規化手段２７、位置／大きさ検出手段１７、パター

ンマッチング手段２９、及び画像保持手段３８、は二次元格子状に配列された配列演算ユニット１００（ARRAY OPERATION UNIT）から構成されるデータ処理装置１１０を用いることにより実装することができる。そこで以下では、配列演算ユニット１００を利用したデータ処理装置１１０の実施形態を挙げ、図面を参照して説明する。

【００１７】まず配列演算ユニット１００は、入力画像の１つの画素とその近傍画素を用いることにより、出力画像の１つの画素を生成する。そこで図７に示したように、配列演算ユニット１００を入力画像のサイズに合わせて二次元格子状に配列したデータ処理装置１１０を用いることにより、データ処理装置１１０は入力画像から出力画像を生成することができる。なお図7において、配列演算ユニット１００をＡＯＵと略記する。したがって配列演算ユニット１００のアルゴリズムを示すことにより、データ処理装置１１０の画像処理を示すことができる。そこで配列演算ユニット１００のアルゴリズムを示すために、画像記憶手段１２、画像振動手段、エッジ情報生成手段１４、エッジ情報形成手段１５、物体／背景分離手段１６、領域正規化手段２７、位置／大きさ検出手段１７、パターンマッチング手段２９、及び画像保持手段３８、で用いる数式について説明する。

【００１８】幅w、高さｈ、帯域数ｂの任意の2^n階調画像を$\underline{x}$、$\underline{y}$、$\underline{w}$とすると、$\underline{x}$、$\underline{y}$、$\underline{w}$は各々位置ｐ（ｉ，ｊ，ｋ）の帯域画素値x_{ijk}、y_{ijk}、w_{ijk}を用いて数式１、２及び３のように表される。なおアンダーラインが付された文字はベクトルを示す。またｎは非負の整数、ｗ、ｈ、ｂ、ｉ、ｊ、ｋは自然数である。

【００１９】

【数１】

$$\underline{x} = \{x_{ijk} | x_{ijk} \text{ is value at } p(i,j,k), 1 \le i \le w, 1 \le j \le h, 1 \le k \le b\}$$

【００２０】

【数２】

$$\underline{y} = \{y_{ijk} | y_{ijk} \text{ is value at } p(i,j,k), 1 \le i \le w, 1 \le j \le h, 1 \le k \le b\}$$

【００２１】

【数３】

$$\underline{w} = \{w_{ijk} | w_{ijk} \text{ is value at } p(i,j,k), 1 \le i \le w, 1 \le j \le h, 1 \le k \le b\}$$

【００２２】まず前記画像の各帯域画素値に対する点処理に関する関数について以下で説明する。

【００２３】画像ｘを二値画像に変換する場合、数式４に従って帯域画素値を二値化する。

【００２４】

【数４】

$$\Phi_{ijk}(\underline{x}) = \begin{cases} 1 & \text{if } x_{ijk} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

【００２５】画像ｘを帯域最大値画像に変換する場合、数式５に従ってｉ行ｊ列の画素の各帯域の値のうち最大値を選択する。なお前記帯域最大値画像は単帯域画像となるので、便宜上帯域数１の前記画像として取り扱うことにする。したがって関数$B_{ij1}(\underline{x})$の第３添字は１となっている。

【００２６】

【数５】

$$B_{ij1}(\underline{x}) = \max_k \{x_{ijk}\}$$

【００２７】画像ｘが二値画像であるとして、画像$\underline{x}$を反転させる場合、数式６に従って計算する。

【００２８】

【数６】

$$I_{ijk}(\underline{x}) = 1 - x_{ijk}$$

【００２９】画像ｘの位置ｐ（ｉ，ｊ，ｋ）における対数変換は数式７に従って行われる。ここでｅはオフセットであり、自然対数関数が出力する値が有効範囲に入るようにするために使われるので、一般にｅ＝１で十分である。この対数化により帯域画素値同士の除算を減算にすることができる。また画像$\underline{x}$が2^n階調のデジタル画像１１１であるとすると、帯域数に関わらず2^n個の要素を含むルックアップテーブルをメモリ１０２上に持つならば、毎回自然対数関数を計算する必要もなく、標準的な対数表を持つ必要もなくなる。

【００３０】

【数７】

$$L_{ijk}(\underline{x}) = \ln(x_{lmk} + e)$$

【００３１】さて、画像の位置ｐ（ｉ，ｊ，ｋ）におけるｑ近傍の位置の集合P_{ijk}（ｑ）は数式８によって表される。ただしｑは４、８、２４、４８、８０、１２０、（２ｒ＋１）^{2}－１と続く数列であり、ｒは自然数である。なお画像サイズをはみ出した位置が集合P_{ijk}（ｑ）に含まれる場合には、特に指定がない限り位置ｐ（ｉ，ｊ，ｋ）を代用するものとする。またこれ以外のときは、指定に従い、画素値が０に相当し、しかも画像に含まれない架空の位置を代用する。これにより辺縁処理は自動的に行われる。したがって集合P_{ijk}（ｑ）の要素の数N_{ijk}は常にｑとなる。

【００３２】

【数８】

$$P_{ijk}(q)=\begin{cases}\{p(i+1,j,k),\ p(i,j+1,k),\ p(i-1,j,k),\ p(i,j-1,k)\} & \text{if } q=4,\\ \{p(l,m,k)\mid i-r\le l\le i+r,\ j-r\le m\le j+r,\ p(l,m,k)\neq p(i,j,k)\} & \text{if } q=(2r+1)^2-1.\end{cases}$$

【００３３】そこで次に画像の各帯域画素値に対する最大８近傍の近傍処理に関する関数及びオペレータについて以下で説明する。

【００３４】画像ｘの位置ｐ（ｉ，ｊ，ｋ）における振動は数式９に従って行われる。ここで位置ｐ（ｉ，ｊ，ｋ）のｑ近傍の中から１つの位置だけを選択する方法によって画像単位で振動させるか画素単位で振動させるかを決定することができる。もし画像ｘの全ての位置において、全く同じ方法によりｑ近傍の中から１つを選択すれば、画像ｘは画像単位で振動する。一方で画像ｘの各々の位置において、疑似乱数などを用いてランダムにｑ近傍の中から１つを選択すれば、画像ｘは画素単位で振動する。

【００３５】

【数９】

$$\Xi_{ijk}(\underline{x})=x_{lmk}\quad\text{for only one of } p(l,m,k)\in P_{ijk}(q)$$

【００３６】画像ｘの位置ｐ（ｉ，ｊ，ｋ）における平滑化は数式１０に従って行われる。ただしｉｎｔ（ｖ）は実数ｖの小数点以下切り捨てを意味するものとする。もし画像ｘの帯域画素値が整数値であるならば、ハードウェアの実装時に$N_{ijk}=4$のときx_{lmk}の総和に対して右シフト命令を２回、$N_{ijk}=8$のときx_{lmk}の総和に対して右シフト命令を３回実行するような回路に変更することにより、除算を実行する回路を省くことができる。

【００３７】

【数１０】

$$S_{ijk}(\underline{x})=\mathrm{int}\left(\frac{1}{N_{ijk}}\sum_{p(l,m,k)\in P_{ijk}(q)}x_{lmk}\right)$$

【００３８】ラプラシアンの計算であるが、これは数式１１に示すように単なる２階差分オペレータである。８近傍の方がノイズの微妙な変化を捉えてゼロ点およびゼロ交差が多くなり、本発明には向いている。ただしN_{ijk}が4か8であるので、もしハードウェアの実装時に$N_{ijk}=4$のときx_{ijk}に対して左シフト命令を２回、$N_{ijk}=8$のときx_{ijk}に対して左シフト命令を３回実行するような回路に変更することにより、実数の乗算を実行する回路を省くことができる。

【００３９】

【数１１】

$$\nabla^2_{ijk}\underline{x}=\sum_{p(l,m,k)\in P_{ijk}(q)}x_{lmk}-N_{ijk}x_{ijk}$$

【００４０】ラプラシアンによって求められた値からゼロ点を見付ける方法として、従来は正から負に変化する画素を見付けていたが、本発明では数式１２に従い、負から正にゼロ交差する画素の他に、負からゼロやゼロから正などゼロ点が経由したり、ゼロが継続する画素を見付けるようにする。本発明では、数式１２が見付けたゼロ点はエッジがある場所ではなく、ノイズがある場所、つまりエッジがない場所になる。また数式１２により実数値の二値化も同時に行っている。

【００４１】

【数１２】

$$Z_{ijk}(\underline{x})=\begin{cases}1 & \text{if } x_{ijk}\le 0 \text{ and } x_{lmk}\ge 0 \text{ for } {}^{\exists}p(l,m,k)\in P_{ijk}(q),\\ 0 & \text{otherwise.}\end{cases}$$

【００４２】画像ｘが任意の二値画像であるとして、画像ｘのうち孔が空いている画素を埋める場合には、数式１３に従い計算する。ここでｆは埋めるべき孔の大きさを表すパラメータであり、一般にはｆ＝１で十分である。なお４近傍の場合にはその性質上対角線を検知することができないので、極力８近傍にした方がよい。

【００４３】

【数１３】

$$F_{ijk}(\underline{x})=\begin{cases}1 & \text{if } \sum_{p(l,m,k)\in P_{ijk}(q)}x_{lmk}+f\ge N_{ijk},\\ x_{ijk} & \text{otherwise.}\end{cases}$$

【００４４】画像ｘが任意の二値画像であるとして、画像ｘのうち孤立点ないし孤立孔を削除する場合には、数式１４に従い計算する。なお４近傍の場合にはその性質上対角線を検知することができないので、極力８近傍にした方がよい。

【００４５】

【数１４】

$$A_{ijk}(\underline{x}) = \begin{cases} 0 & \text{if } x_{ijk} = 1 \text{ and } \sum_{p(l,m,k)\subset P_{ijk}(q)} x_{lmk} = 0, \\ 1 & \text{if } x_{ijk} = 0 \text{ and } \sum_{p(l,m,k)\subset P_{ijk}(q)} x_{lmk} = N_{ijk}, \\ x_{ijk} & \text{otherwise.} \end{cases}$$

【００４６】画像ｘが任意の二値画像であるとして、画像ｘのうち線幅が１である画素を検知するために、４近傍画素を用いて数式１５に従い計算する。

【００４７】

【数１５】

$$J_{ijk}(\underline{x}) = \begin{cases} x_{ijk} & \text{if } x_{i-1jk} + x_{i+1jk} = 0 \text{ or } x_{ij-1k} + x_{ij+1k} = 0, \\ 0 & \text{otherwise.} \end{cases}$$

【００４８】２つの画像ｘ、ｙが任意の二値画像であり、画像ｙが画像ｘのうち線幅が１である画素を検知した画像であるとすると、画像ｘのうち線幅が１である画素の線幅を拡張するために、４近傍画素を用いて数式１６に従い計算する。

【００４９】

【数１６】

$$K_{ijk}(\underline{x}, \underline{y}) = \begin{cases} 1 & \text{if } y_{i-1jk} + y_{i+1jk} + y_{ij-1k} + y_{ij+1k} > 0, \\ x_{ijk} & \text{otherwise.} \end{cases}$$

【００５０】そこで数式１５の線幅検知と数式１６の線幅拡張を用いると、数式１７に従い二値画像の線幅補完を簡単に記述することができる。

【００５１】

【数１７】

$$C_{ijk}(\underline{x}) = K_{ijk}(\underline{x}, \underline{J(x)})$$

【００５２】次に画像の各帯域画素値に対する近傍処理に関する関数及びオペレータについて以下で説明する。

【００５３】２つの画像ｘ、ｙがある場合、これらの画像の最大値画像は数式１８に従って計算される。

【００５４】

【数１８】

$$M_{ijk}(\underline{x}, \underline{y}) = \begin{cases} x_{ijk} & \text{if } x_{ijk} \geq y_{ijk}, \\ y_{ijk} & \text{otherwise.} \end{cases}$$

【００５５】２つの画像ｘ、ｙがある場合、これらの画像の差分は数式１９に従って計算される。

【００５６】

【数１９】

$$D_{ijk}(\underline{x}, \underline{y}) = x_{ijk} - y_{ijk}$$

【００５７】ここで数式１１のラプラシアンと数式１９の差分を用いると、数式２０に従い画像の鮮鋭化を簡単に記述することができる。

【００５８】

【数２０】

$$E_{ijk}(\underline{x}) = D_{ijk}(\underline{x}, \nabla^2_{ijk}\underline{x})$$

【００５９】２つの画像ｘ、ｙがあり、画像ｙが単帯域二値画像である場合、数式２１に従い、画像ｙの帯域画素値を用いて画像ｘの各帯域画素値をマスクすることができる。

【００６０】

【数２１】

$$O_{ijk}(\underline{x}, \underline{y}) = x_{ijk}y_{ij1}$$

【００６１】２つの画像ｘ、ｙがあり、画像ｘとｙが二値画像である場合、数式２２に従い、画像ｘを基に画像ｙを整形することができる。

【００６２】

【数２２】

$$Q_{ijk}(\underline{x}, \underline{y}) = \begin{cases} x_{ijk} & \text{if } y_{ijk} + \sum_{p(l,m,k)\in P_{ijk}(q)} y_{lmk} > 0, \\ 0 & \text{otherwise.} \end{cases}$$

【００６３】２つの画像ｘ、ｙがあり、画像ｙが二値画像である場合、数式２３に従い、画像ｙで指定されなかった画像ｘの帯域画素値を、画像ｘの帯域画素値の近傍のうち画像ｙで指定された画像ｘの帯域画素値の平均値で補間する。ただしｉｎｔ（ｖ）は実数ｖの小数点以下切り捨てを意味するものとする。

【００６４】

【数２３】

$$V_{ijk}(\underline{x}, \underline{y}) = \begin{cases} \text{int}\left(\frac{\sum_{p(l,m,1)\in P_{ij1}(q)} x_{lmk}y_{lm1}}{\sum_{p(l,m,1)\in P_{ij1}(q)} y_{lm1}}\right) & \text{if } y_{ij1} = 0 \text{ and } \sum_{p(l,m,1)\in P_{ij1}(q)} y_{lm1} > 0, \\ x_{ijk} & \text{otherwise.} \end{cases}$$

【００６５】さて本発明では、画素の位置や移動量なども画像データのように扱うことで処理を単純にしてい

る。これを位置の画像化と呼ぶ。以下では画像化に関する幾つかの関数及びオペレータについて説明する。

【0066】まず位置p（l，m，o）のl、m、o各々の値を画像データとして帯域画素値に変換するオペレータを#とし、変換された帯域画素値を#p（l，m，o）とする。次に帯域画素値が位置p（i，j，k）から位置p（i+l，j+m，k+o）へ移動する場合を考える。このとき帯域画素値の移動量は位置p（l，m，o）として表されるものとする。つまり移動量をある位置からのベクトルと見なすことができる。最後に帯域画素値から位置を取り出すオペレータを$\#^{-1}$とする。したがって$\#^{-1}\#p(l,m,o)=p(l,m,o)$となる。

【0067】そこで数式24に従い、移動量p（i，j，k）を幅方向と高さ方向で表される平面内で180度反対方向に向けることができる。

【0068】

【数24】

$$\Upsilon(p(i,j,k))=p(-i,-j,k)$$

【0069】画像xがあり、画像xが単帯域二値画像である場合、画像xの位置p（i，j，1）における重心位置への移動量は数式25に従い計算される。なお、本来重心を計算する際には除算を行う必要があるが、8近傍内への移動量を計算する際に除算は相殺されてしまうので、数式25では除算が省かれている。

【0070】

【数25】

$$G_{ij1}(\underline{x})=p\Big(\sum_{p(l,m,1)\in P_{ij1}(q)}(l-i)x_{lm1},\ \sum_{p(l,m,1)\in P_{ij1}(q)}(m-j)x_{lm1},0\Big)$$

【0071】移動量p（i，j，k）から、数式26及び27に従い8近傍内への移動量を計算し、移動量画像に画像化することができる。なお数式27は、画像の離散化により数式26では対応しきれない場合にのみ利用する。

【0072】

【数26】

$$\Theta(p(i,j,k))=\begin{cases}\#p(1,0,k) & \text{if } i>0,\ |j|<|i|/2,\\ \#p(1,-1,k) & \text{if } i>0,\ j<0,\ |i|/2\le|j|\le 2|i|,\\ \#p(0,-1,k) & \text{if } j<0,\ 2|i|<|j|,\\ \#p(-1,-1,k) & \text{if } i<0,\ j<0,\ |i|/2\le|j|\le 2|i|,\\ \#p(-1,0,k) & \text{if } i<0,\ |j|<|i|/2,\\ \#p(-1,1,k) & \text{if } i<0,\ j>0,\ |i|/2\le|j|\le 2|i|,\\ \#p(0,1,k) & \text{if } j>0,\ 2|i|<|j|,\\ \#p(1,1,k) & \text{if } i>0,\ j>0,\ |i|/2\le|j|\le 2|i|,\\ \#p(0,0,k) & \text{otherwise.}\end{cases}$$

【0073】

【数27】

$$\Theta'(p(i,j,k))=\begin{cases}\#p(1,0,k) & \text{if } i>0,\ |j|<|i|/2,\\ \#p(1,0,k) & \text{if } i>0,\ j<0,\ |i|/2\le|j|\le 2|i|,\\ \#p(0,1,k) & \text{if } i<0,\ j>0,\ |i|/2\le|j|\le 2|i|,\\ \#p(0,1,k) & \text{if } j>0,\ 2|i|<|j|,\\ \#p(1,1,k) & \text{if } i>0,\ j>0,\ |i|/2\le|j|\le 2|i|,\\ \#p(0,0,k) & \text{otherwise.}\end{cases}$$

【0074】したがって数式25、26及び27を用いると、数式28及び29に従い、単帯域二値画像xの重心方向への移動量画像の帯域画素値を簡単に記述することができる。なお移動量画像の帯域数は1となる。

【0075】

【数28】

$$\Lambda_{ij1}(\underline{x})=\Theta(G_{ij1}(\underline{x}))$$

【0076】

【数29】

$$\Lambda'_{ij1}(\underline{x})=\Theta'(G_{ij1}(\underline{x}))$$

【0077】一方で数式24を用いると重心位置の反対位置を求めることができるので、数式30に従い、単帯域二値画像xの重心と反対方向への移動量画像の帯域画素値を簡単に記述することができる。なお移動量画像の帯域数は1となる。

【0078】

【数30】

$$R_{ij1}(\underline{x})=\Theta(\Upsilon(G_{ij1}(\underline{x})))$$

【0079】2つの画像x、yがあり、画像yが移動量画像である場合、数式31に従い、画像yで指し示された移動位置に画像xの帯域画素値を移動した後、同じ帯

域画素に移動した帯域画素値の合計を濃淡画像にすることができる。

【0080】

【数31】

$$\Gamma_{ijk}(\underline{x},\underline{y}) = \sum x_{lmk} \quad \text{for } p(l,m,1) \in P_{ij1}(q) \text{ and } \sharp^{-1}y_{lm1} = p(i-l,j-m,0)\ .$$

【0081】そこで数式4、28、29及び31を用いることにより、数式32又は数式33に従い、単帯域濃淡画像xを近傍の重心方向に移動した後、同じ帯域画素に移動した帯域画素値の合計を簡単に記述することができる。

【0082】

【数32】

$$\Lambda_{ij1}(\underline{x}) = \Gamma_{ij1}(\underline{x},\underline{\Delta}(\underline{\Phi}(\underline{x})))$$

【0083】

【数33】

$$\Lambda'_{ij1}(\underline{x}) = \Gamma_{ij1}(\underline{x},\underline{\Delta}'(\underline{\Phi}(\underline{x})))$$

【0084】2つの画像x、yがあり、画像xが二値画像で、画像yが移動量画像である場合、画像xの各帯域画素値の移動先の位置を求めることができるので、移動先が重複する帯域画素値を見つけることができる。そこで画像xの各帯域画素値の移動先が重複することなく、しかも移動する各帯域画素値が存在することを表す移動可能画像の帯域画素値は、数式34に従い生成される。なお移動可能画像の帯域数は1となる。

【0085】

【数34】

$$H_{ij1}(\underline{x},\underline{y}) = \begin{cases} 1 & \text{if } x_{ij1} = 0 \text{ and } \sharp^{-1}y_{lm1} = p(i-l,j-m,0) \\ & \text{for only one of } p(l,m,1) \in P_{ij1}(q), \\ 0 & \text{otherwise.} \end{cases}$$

【0086】3つの画像x、y、wがあり、画像yが移動可能画像であり、画像wが移動量画像である場合、数式35に従い画像xの帯域画素値を移動することができる。

【0087】

【数35】

$$T_{ijk}(\underline{x},\underline{y},\underline{w}) = \begin{cases} x_{lmk} & \text{if } y_{ij1} = 1 \text{ and } \sharp^{-1}w_{lm1} = p(i-l,j-m,0) \\ & \text{for } {}^{\exists}p(l,m,1) \in P_{ij1}(q), \\ 0 & \text{if } y_{lm1} = 1 \text{ and } \sharp^{-1}w_{ij1} = p(l-i,m-j,0) \\ & \text{for } {}^{\exists}p(l,m,1) \in P_{ij1}(q), \\ x_{ijk} & \text{otherwise.} \end{cases}$$

【0088】そこで数式30、数式34及び数式35を用いると、数式36に従い、二値画像yから計算される重心位置と反対方向に画像xの帯域画素を移動することで得られる画像の帯域画素値を簡単に記述することができる。

【0089】

【数36】

$$U_{ijk}(\underline{x},\underline{y}) = T_{ijk}(\underline{x},\underline{H}(\underline{y},\underline{R}(\underline{y})),\underline{R}(\underline{y}))$$

【0090】そこで数式1から数式36までを用いることにより、画像記憶手段12、画像振動手段、エッジ情報生成手段14、エッジ情報形成手段15、領域正規化手段27、位置／大きさ検出手段17、及び画像保持手段38、を実装するデータ処理装置110の全ての配列演算ユニット100のアルゴリズムを記述することができる。以下では、データ処理装置110中の任意の配列演算ユニット100のアルゴリズムを用いて、画像記憶手段12、画像振動手段、エッジ情報生成手段14、エッジ情報形成手段15、領域正規化手段27位置／大きさ検出手段17、及び画像保持手段38、を説明する。

【0091】データ処理装置110によって実現される画像記憶手段12がデジタル画像111を記憶するために、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上i行j列に配置された配列演算ユニット100をAOU$_{i\ j}$とすると、AOU$_{i\ j}$のアルゴリズムは図8のようになる。

【0092】ステップ1201で、AOU$_{i\ j}$を格子上のi行j列に配置する。これは論理的であれ物理的であれ、AOU$_{i\ j}$の近傍を決定するために必要である。

【0093】ステップ1202で、AOU$_{i\ j}$の近傍や変数の初期値を設定する。

【0094】ステップ1203で、順次入力されるデジタル画像111が無くなったかどうか判断する。もしデジタル画像111が無ければ（ステップ1203：YES）、アルゴリズムを終了する。もしデジタル画像111があれば（ステップ1203：NO）、ステップ1204に移行する。ただし特定の画像サイズのみに対して配列演算ユニット100を実装する場合には、無限ループにしても良い。

【0095】ステップ1204で、デジタル画像111が準備されるまで入力待ちをする。

【0096】ステップ1205で、デジタル画像111

のｉ行ｊ列の画素を帯域数分入力する。このためＡＯＵ$_{ij}$は少なくとも帯域数分の画像データを記憶するメモリ１０２を必要とする。

【００９７】ステップ１２０６で、入力待ちの間出力できるように、デジタル画像１１１のｉ行ｊ列の画素を記憶する。

【００９８】ステップ１２０７で、デジタル画像１１１の帯域画素値を出力する。その後ステップ１２０３に戻る。

【００９９】これにより、配列演算ユニット１００から構成されるデータ処理装置１１０を用いて、画像記憶手段１２はデジタル画像１１１を記憶することができる。

【０１００】データ処理装置１１０によって実現される画像振動手段がデジタル画像１１１を振動させるために、二次元格子状に配列された配列演算ユニット１００は同期して並列に動作する。格子上ｉ行ｊ列に配置された配列演算ユニット１００をＡＯＵ$_{ij}$とすると、ＡＯＵ$_{ij}$のアルゴリズムは図９のようになる。

【０１０１】ステップ１３０１で、ＡＯＵ$_{ij}$を格子上のｉ行ｊ列に配置する。これは論理的であれ物理的であれ、ＡＯＵ$_{ij}$の近傍を決定するために必要である。

【０１０２】ステップ１３０２で、ＡＯＵ$_{ij}$の近傍や変数の初期値を設定する。

【０１０３】ステップ１３０３で、順次入力されるデジタル画像１１１が無くなったかどうか判断する。もしデジタル画像１１１が無ければ（ステップ１３０３：ＹＥＳ）、アルゴリズムを終了する。もしデジタル画像１１１があれば（ステップ１３０３：ＮＯ）、ステップ１３０４に移行する。ただし特定の画像サイズのみに対して配列演算ユニット１００を実装する場合には、無限ループにしても良い。

【０１０４】ステップ１３０４で、デジタル画像１１１のｉ行ｊ列の画素を帯域数分入力する。このためＡＯＵ$_{ij}$は少なくとも帯域数分の画像データを記憶するメモリ１０２を必要とする。

【０１０５】ステップ１３０５で、関数$\Xi_{ijk}(\underline{x})$に従いデジタル画像１１１のｉ行ｊ列の画素を近傍画素の１つに移動させる。

【０１０６】ステップ１３０６で、デジタル画像１１１の帯域画素値を出力する。その後ステップ１３０３に戻る。

【０１０７】これにより、配列演算ユニット１００から構成されるデータ処理装置１１０を用いて、画像振動手段はデジタル画像１１１を振動させることができる。

【０１０８】データ処理装置１１０によって実現されるエッジ情報生成手段１４がデジタル画像１１１から粗エッジ情報画像１１３を生成するために、二次元格子状に配列された配列演算ユニット１００は同期して並列に動作する。格子上ｉ行ｊ列に配置された配列演算ユニット１００をＡＯＵ$_{ij}$とすると、エッジ情報生成手段１４に対するＡＯＵ$_{ij}$のアルゴリズムは図１０のようになる。

【０１０９】ステップ１４０１で、ＡＯＵ$_{ij}$を格子上のｉ行ｊ列に配置する。これは論理的であれ物理的であれ、ＡＯＵ$_{ij}$の近傍を決定するために必要である。

【０１１０】ステップ１４０２で、ＡＯＵ$_{ij}$の近傍や変数の初期値を設定する。近傍の設定においては、前記各関数で使う近傍サイズｑを個別に４か８に決めても良いし、全部を４か８に統一しても良い。本発明のエッジ情報生成手段１４が生成する粗エッジ情報１１２の正確さを上げるためには近傍サイズｑを全て８に設定することが望ましい。しかしながら粗エッジ情報１１２を生成するための計算時間の制約や、デジタル画像１１１の帯域数などにより、エッジ情報生成手段１４は必要に応じて適宜近傍サイズを変えることで対処することができる。

【０１１１】ステップ１４０３で、デジタル画像１１１が終了したかどうか判断する。もしデジタル画像１１１が無ければ（ステップ１４０３：ＹＥＳ）、アルゴリズムを終了する。もしデジタル画像１１１があれば（ステップ１４０３：ＮＯ）、アルゴリズムを終了する。ただし特定の帯域数と画像サイズに対して配列演算ユニット１００を実装する場合には、無限ループにしても良い。

【０１１２】ステップ１４０４で、デジタル画像１１１のｉ行ｊ列の画素を帯域数分入力する。これは、ＡＯＵ$_{ij}$がデジタル画像１１１のｉ行ｊ列の画素を一括して処理するためである。このためＡＯＵ$_{ij}$は少なくとも帯域数分の画像データを記憶するメモリ１０２を必要とする。

【０１１３】ステップ１４０５で、ＡＯＵ$_{ij}$が近傍の配列演算ユニット１００と通信することにより、入力したデジタル画像１１１の各帯域画素値に対して関数$S_{ijk}(\underline{x})$に従い平滑化を行う。平滑化された帯域画素値は平滑化画像の帯域画素値として扱われる。ここで関数$S_{ijk}(\underline{x})$は必要に応じて数回繰り返しても良い。一般的な多帯域画像の場合、この回数は２回で十分である。

【０１１４】ステップ１４０６で、平滑化画像の各帯域画素値に対して関数$L_{ijk}(\underline{x})$に従い対数変換を行う。対数変換された帯域画素値は対数変換画像の帯域画素値として扱われる。

【０１１５】ステップ１４０７で、ＡＯＵ$_{ij}$が近傍の配列演算ユニット１００と通信することにより、対数変換画像の各帯域画素値に対して関数$E_{ijk}(\underline{x})$に従い鮮鋭化を行う。鮮鋭化された帯域画素値は鮮鋭化画像の帯域画素値として扱われる。

【０１１６】ステップ１４０８で、鮮鋭化画像の各帯域画素値に対して関数$D_{ijk}(\underline{x}, y)$に従い１入力前鮮鋭化画像の各帯域画素値を引く。差分を計算された帯域画素値は時間差分画像の帯域画素値として扱われる。

【0117】ステップ1409で、1入力前鮮鋭化画像の各帯域画素値を鮮鋭化画像の対応する各帯域画素値で置き換える。

【0118】ステップ1410で、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、時間差分画像の各帯域画素値に対してオペレータ$\nabla^2{}_{ijk}\underline{x}$に従いラプラシアンの計算を行う。ラプラシアンを計算された帯域画素値は時間差分ラプラシアン画像の帯域画素値として扱われる。

【0119】ステップ1411で、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、時間差分ラプラシアン画像の各帯域画素値に対して関数$Z_{ijk}(\underline{x})$に従いゼロ点を抽出する。ゼロ点を抽出された帯域画素値は時間差分ゼロ点画像の帯域画素値として扱われる。

【0120】ステップ1412で、時間差分ラプラシアン画像の各帯域画素値に対して関数$B_{ij1}(\underline{x})$に従い各帯域画素値のうち最大値を検出する。検出された最大値帯域画素値は最大値時間差分ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。

【0121】ステップ1413で、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、鮮鋭化画像の各帯域画素値に対してオペレータ$\nabla^2{}_{ijk}\underline{x}$に従いラプラシアンの計算を行う。ラプラシアンを計算された帯域画素値はラプラシアン画像の帯域画素値として扱われる。

【0122】ステップ1414で、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、ラプラシアン画像の各帯域画素値に対して関数$Z_{ijk}(\underline{x})$に従いゼロ点を抽出する。ゼロ点を抽出された帯域画素値はゼロ点画像の帯域画素値として扱われる。

【0123】ステップ1415で、ラプラシアン画像の各帯域画素値に対して関数$B_{ij1}(\underline{x})$に従い各帯域画素値のうち最大値を検出する。検出された最大帯域画素値は最大値ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。

【0124】ステップ1416で、ラプラシアン画像の各帯域画素値と時間差分ラプラシアン画像の各帯域画素値に対して関数$M_{ijk}(\underline{x}, y)$に従い各々の画像の同じ位置にある帯域画素値のうち最大値を検出する。検出された最大帯域画素値は混成ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。

【0125】ステップ1417で、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、混成ゼロ点画像の帯域画素値に対して関数$F_{ijk}(\underline{x})$に従い孔を除去する。孔を除去された帯域画素値は孔除去混成ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。ここで関数$F_{ijk}(\underline{x})$は必要に応じて数回繰り返しても良い。一般的な多帯域画像の場合、この回数は1回で十分である。

【0126】ステップ1418で、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、孔除去混成ゼロ点画像の帯域画素値に対して関数$A_{ijk}(\underline{x})$に従い孤立点および孤立孔を除去する。孤立点および孤立孔を除去された帯域画素値はノイズ除去混成ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。

【0127】ステップ1419で、ノイズ除去混成ゼロ点画像の帯域画素値に対して関数$I_{ijk}(\underline{x})$に従い0と1を反転させる。反転された帯域画素値は粗エッジ情報画像113の帯域画素値として扱われる。

【0128】ステップ1420で、粗エッジ情報画像113の帯域画素値を出力する。その後ステップ1403に戻る。

【0129】これにより、配列演算ユニット100から構成されるデータ処理装置110を用いて、エッジ情報生成手段14はデジタル画像111から粗エッジ情報画像113を生成することができる。

【0130】図11に示すように、データ処理装置110によって実現されるエッジ情報形成手段15が粗エッジ情報112から構成される粗エッジ情報画像113、及びデジタル画像111から、形成エッジ情報114から構成される形成エッジ情報画像115を生成するために、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上i行j列に配置された配列演算ユニット100をAOU_{ij}とすると、AOU_{ij}のアルゴリズムは図12のようになる。

【0131】ステップ1501で、AOU_{ij}を格子上のi行j列に配置する。これは論理的であれ物理的であれ、AOU_{ij}の近傍を決定するために必要である。

【0132】ステップ1502で、AOU_{ij}の近傍や変数の初期値を設定する。近傍の設定においては、前記各関数で使う近傍サイズqを個別に4か8に決めても良いし、全部を4か8に統一しても良い。本発明のエッジ情報形成手段15が形成した形成エッジ情報114の正確さを上げるためには近傍サイズqを全て8に設定することが望ましい。しかしながら粗エッジ情報112を形成するための計算時間の制約や、入力されるデジタル画像111の帯域数などにより、エッジ情報形成手段15は必要に応じて適宜近傍サイズを変えることで対処することができる。

【0133】ステップ1503で、順次入力されるデジタル画像111又は粗エッジ情報画像113が無くなったかどうか判断する。もしデジタル画像111若しくは粗エッジ情報画像113のいずれかが無ければ(ステップ1503:YES)、アルゴリズムを終了する。もしデジタル画像111若しくは粗エッジ情報画像113のいずれかがあれば(ステップ1503:NO)、ステップ1504に移行する。ただし特定の帯域数と画像サイズに対して配列演算ユニット100を実装する場合に

は、無限ループにしても良い。

【0134】ステップ1504で、デジタル画像111及び粗エッジ情報画像113のi行j列の画素を帯域数分入力する。これは、AOU_{ij}がデジタル画像111及び粗エッジ情報画像113のi行j列の画素を一括して処理するためである。このためAOU_{ij}は少なくとも帯域数分の画像データを記憶するメモリ102を必要とする。

【0135】ステップ1505で、デジタル画像111のi行j列の画素と粗エッジ情報画像113のi行j列の画素を分離する。これは、AOU_{ij}がデジタル画像111のi行j列の画素と粗エッジ情報画像113のi行j列の画素を各々独立した画像の画素として処理するためである。もしデジタル画像111のi行j列の画素と粗エッジ情報画像113のi行j列の画素が初めから分離されて入力されていれば、特に何もしない。

【0136】ステップ1506で、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、入力したデジタル画像111の各帯域画素値に対して関数$S_{ijk}(\underline{x})$に従い平滑化を行う。平滑化された帯域画素値は平滑化画像の帯域画素値として扱われる。ここで関数$S_{ijk}(\underline{x})$は必要に応じて数回繰り返しても良い。一般的な多帯域画像の場合、この回数は2回で十分である。

【0137】ステップ1507で、平滑化画像の各帯域画素に対して関数$L_{ijk}(\underline{x})$に従い対数変換を行う。対数変換された帯域画素値は対数変換画像の帯域画素値として扱われる。

【0138】ステップ1508で、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、対数変換画像の各帯域画素値に対して関数$E_{ijk}(\underline{x})$に従い鮮鋭化を行う。鮮鋭化された帯域画素値は鮮鋭化画像の帯域画素値として扱われる。

【0139】ステップ1509で、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、鮮鋭化画像の各帯域画素値に対してオペレータ$\nabla^2{}_{ijk}\underline{x}$に従いラプラシアンの計算を行う。ラプラシアンを計算された帯域画素値はラプラシアン画像の帯域画素値として扱われる。

【0140】ステップ1510で、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、ラプラシアン画像の各帯域画素値に対して関数$Z_{ijk}(\underline{x})$に従いゼロ点を抽出する。ゼロ点を抽出された帯域画素値はゼロ点画像の帯域画素値として扱われる。

【0141】ステップ1511で、ゼロ点画像の各帯域画素値に対して関数$B_{ij1}(\underline{x})$に従い各帯域画素値のうち最大値を検出する。検出された最大帯域画素値は最大値ゼロ点画像の帯域画素値として扱われる。なお便宜上帯域数は1である。

【0142】ステップ1512で、最大値ゼロ点画像の帯域画素値に対して関数$I_{ijk}(\underline{x})$に従い0と1を反転させる。反転された帯域画素値は基礎エッジ情報画像の帯域画素値として扱われる。

【0143】ステップ1513で、入力した粗エッジ情報画像113の帯域画素値は初め整形粗エッジ情報画像の帯域画素値として扱われ、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、基礎エッジ情報画像の帯域画素値を用いて、整形粗エッジ情報画像の帯域画素値に対して関数$Q_{ijk}(\underline{x},\underline{y})$に従い整形を行う。整形された帯域画素値は再び整形粗エッジ情報画像の帯域画素値として扱われる。ここで関数$Q_{ijk}(\underline{x},\underline{y})$は本来整形粗エッジ情報画像の帯域画素値が変化しなくなるまで繰り返される。ただし計算時間の制約、入力される粗エッジ情報画像113の品質、形成される形成エッジ情報画像115に求められる品質などにより、整形処理は適当な繰り返し回数で計算を打ち切った方が良い。

【0144】ステップ1514で、AOU_{ij}が近傍の配列演算ユニット100と通信することにより、整形粗エッジ情報画像の帯域画素値に対して関数$C_{ijk}(\underline{x})$に従い線幅補完を行う。補完された帯域画素値は形成エッジ情報画像115の帯域画素値として扱われる。

【0145】ステップ1515で、形成エッジ情報画像115の帯域画素値を出力する。その後ステップ1503に戻る。

【0146】これにより、配列演算ユニット100から構成されるデータ処理装置110を用いて、エッジ情報形成手段15は粗エッジ情報画像113を形成エッジ情報画像115に形成することができる。

【0147】図13に示すように、データ処理装置110によって実現される領域正規化手段27が物体領域141を含む物体領域画像142、及び物体領域141と重なる分離物体領域143を含むデジタル画像111から正規化領域144を含む正規化画像145を生成するために、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上i行j列に配置された配列演算ユニット100をAOU_{ij}とすると、AOU_{ij}のアルゴリズムは図14のようになる。

【0148】ステップ2701で、AOU_{ij}を格子上のi行j列に配置する。これは論理的であれ物理的であれ、AOU_{ij}の近傍を決定するために必要である。

【0149】ステップ2702で、AOU_{ij}の近傍や変数の初期値を設定する。近傍の設定においては、前記各関数で使う近傍サイズqを個別に決めても良いし、全部を統一しても良い。本発明の領域正規化手段27が生成した正規化画像145の正確さを上げるためには近傍サイズqを全て大きな値に設定することが望ましい。しかしながら分離物体領域143を正規化するための計算時間の制約や、入力されるデジタル画像111のサイズ

などにより、領域正規化手段２７は必要に応じて適宜近傍サイズを変えることで対処することができる。

【０１５０】ステップ２７０３で、順次入力される物体領域画像１４２又はデジタル画像１１１が無くなったかどうか判断する。もし物体領域画像１４２又はデジタル画像１１１が無ければ（ステップ２７０３：ＹＥＳ）、アルゴリズムを終了する。もし物体領域画像１４２又はデジタル画像１１１があれば（ステップ２７０３：ＮＯ）、ステップ２７０４に移行する。ただし特定の帯域数及び画像サイズのみに対して配列演算ユニット１００を実装する場合には、無限ループにしても良い。

【０１５１】ステップ２７０４で、物体領域画像１４２のｉ行ｊ列の画素を１帯域分と、デジタル画像１１１のｉ行ｊ列の画素を帯域数分を入力する。これは、AOU_{ij}が物体領域画像１４２のｉ行ｊ列の画素とデジタル画像１１１のｉ行ｊ列の画素を一括して処理するためである。このためAOU_{ij}は少なくとも総帯域数分の画像データを記憶するメモリ１０２を必要とする。

【０１５２】ステップ２７０５で、物体領域画像１４２のｉ行ｊ列の画素とデジタル画像１１１のｉ行ｊ列の画素を分離する。これは、AOU_{ij}が物体領域画像１４２のｉ行ｊ列の画素とデジタル画像１１１のｉ行ｊ列の画素を各々独立した画像の画素として処理するためである。もし物体領域画像１４２のｉ行ｊ列の画素とデジタル画像１１１のｉ行ｊ列の画素が初めから分離されて入力されていれば、特に何もしない。物体領域画像１４２及びデジタル画像１１１は各々更新物体領域画像及び更新デジタル画像にコピーされる。

【０１５３】ステップ２７０６で、AOU_{ij}が近傍の配列演算ユニット１００と通信することにより、更新物体領域画像の各帯域画素値に対して関数$R_{ij1}(\underline{x})$に従い移動量を計算する。移動量を画像化した帯域画素値は移動量画像の帯域画素値として扱われる。

【０１５４】ステップ２７０７で、AOU_{ij}が近傍の配列演算ユニット１００と通信することにより、更新物体領域画像の各帯域画素値に対して関数$H_{ijk}(\underline{x},\underline{y})$に従い移動可能な移動先帯域画素値を見つけることができる。移動可能な移動先であるかどうかを表す値は移動可能画像の帯域画素値として扱われる。

【０１５５】ステップ２７０８で、AOU_{ij}が近傍の配列演算ユニット１００と通信することにより、更新物体領域画像の各帯域画素値に対して関数$U_{ijk}(\underline{x},\underline{y})$に従い移動可能先に移動させる。移動した帯域画素値は新たに更新物体領域画像の帯域画素値として扱われる。

【０１５６】ステップ２７０９で、AOU_{ij}が近傍の配列演算ユニット１００と通信することにより、更新デジタル画像の各帯域画素値に対して関数$U_{ijk}(\underline{x},\underline{y})$に従い移動可能先に移動させる。移動した帯域画素値は新たに更新デジタル画像の帯域画素値として扱われる。

【０１５７】ステップ２７１０で、ステップ２７０６からステップ２７０９までの繰り返し回数を表す移動回数が指定回数に達したかどうか判断する。もし移動回数が指定回数に達していなければ（ステップ２７１０：ＮＯ）、ステップ２７０６に戻る。もし移動回数が指定回数に達していれば（ステップ２７１０：ＹＥＳ）、ステップ２７１１に移行する。なおこの指定回数はデジタル画像１１１のサイズやデジタル画像１１１の分離物体領域１４３のサイズ、さらには近傍のサイズｑにより決定される。利用目的に応じて適切なパラメータを設定すれば、指定回数を大目に決定しても問題はないが、あまり指定回数を多くしすぎると、正規化に要する時間が長くなる。

【０１５８】ステップ２７１１で、AOU_{ij}が近傍の配列演算ユニット１００と通信することにより、移動を完了した更新物体領域画像の各帯域画素値に対して関数$V_{ijk}(x,y)$に従い近傍の平均値で補間する。なお$\underline{x}$と$\underline{y}$は共に更新物体領域画像となる。平均値で埋められた帯域画素値は正規化された更新物体領域画像の帯域画素値として扱われる。

【０１５９】ステップ２７１２で、AOU_{ij}が近傍の配列演算ユニット１００と通信することにより、移動を完了した更新デジタル画像の各帯域画素値に対して関数$V_{ijk}(x,y)$に従い近傍の平均値で埋める。なお$\underline{x}$は更新デジタル画像となり、$\underline{y}$は更新物体領域画像となる。平均値で埋められた帯域画素値は正規化された更新デジタル画像の帯域画素値として扱われる。

【０１６０】ステップ２７１３で、ステップ２７１１からステップ２７１２までの繰り返し回数を表す補間回数が指定回数に達したかどうか判断する。もし補間回数が指定回数に達していなければ（ステップ２７１３：ＮＯ）、ステップ２７１１に戻る。もし補間回数が指定回数に達していれば（ステップ２７１３：ＹＥＳ）、ステップ２７１４に移行する。一般的に補間回数は近傍サイズｑの半分程度の回数で十分である。

【０１６１】ステップ２７１４で、ステップ２７０６からステップ２７１３までの繰り返し回数を表す継続回数が指定回数に達したかどうか判断する。もし継続回数が指定回数に達していなければ（ステップ２７１４：ＮＯ）、ステップ２７０６に戻る。もし継続回数が指定回数に達していれば（ステップ２７１４：ＹＥＳ）、ステップ２７１５に移行する。なおこの指定回数はデジタル画像１１１のサイズやデジタル画像１１１の分離物体領域１４３のサイズ、さらには近傍のサイズｑにより決定される。利用目的に応じて適切なパラメータを設定すれば、指定回数を大目に決定しても問題はないが、あまり指定回数を多くしすぎると、正規化に要する時間が長くなる。

【０１６２】ステップ２７１５で、更新デジタル画像の

帯域画素値を正規化画像１４５の帯域画素値として出力する。その後ステップ２７０３に戻る。

【０１６３】これにより、配列演算ユニット１００から構成されるデータ処理装置１１０を用いて、領域正規化手段２７が物体領域画像１４２及びデジタル画像１１１から正規化画像１４５を生成することができる。

【０１６４】図１５に示すように、データ処理装置１１０によって実現される位置／大きさ検出手段１７が粗エッジ情報１１２を画素とする粗エッジ情報画像１１３から重複情報１３１を画素とする重複情報画像１３２を生成するために、二次元格子状に配列された配列演算ユニット１００は同期して並列に動作する。格子上ｉ行ｊ列に配置された配列演算ユニット１００をAOU_{ij}とすると、AOU_{ij}のアルゴリズムは図１６のようになる。

【０１６５】ステップ１７０１で、AOU_{ij}を格子上のｉ行ｊ列に配置する。これは論理的であれ物理的であれ、AOU_{ij}の近傍を決定するために必要である。

【０１６６】ステップ１７０２で、AOU_{ij}の近傍や変数の初期値を設定する。近傍の設定においては、前記各関数で使う近傍サイズｑを個別に決めても良いし、全部を統一しても良い。本発明のデータ処理装置１１０が生成した重複情報画像１３２の正確さを上げるためには近傍サイズｑを全て大きな値に設定することが望ましい。しかしながら物体の粗エッジ情報１１２の重心を計算するための計算時間の制約や、入力される粗エッジ情報画像１１３のサイズなどにより、位置／大きさ検出手段１７は必要に応じて適宜近傍サイズを変えることで対処することができる。

【０１６７】ステップ１７０３で、順次入力される粗エッジ情報画像１１３が無くなったかどうか判断する。もし粗エッジ情報画像１１３が無ければ（ステップ１７０３：ＹＥＳ）、アルゴリズムを終了する。もし粗エッジ情報画像１１３があれば（ステップ１７０３：ＮＯ）、ステップ１７０４に移行する。ただし特定の画像サイズのみに対して配列演算ユニット１００を実装する場合には、無限ループにしても良い。

【０１６８】ステップ１７０４で、粗エッジ情報画像１１３のｉ行ｊ列の画素を１帯域分入力する。このためAOU_{ij}は少なくとも１帯域分の画像データを記憶するメモリ１０２を必要とする。

【０１６９】ステップ１７０５で、粗エッジ情報画像１１３の粗エッジ情報１１２を重複情報画像１３２の重複情報１３１に変換する。重複情報１３１は１か０に相当する帯域画素値となる。

【０１７０】ステップ１７０６で、AOU_{ij}が近傍の配列演算ユニット１００と通信することにより、重複情報画像１３２の各帯域画素値に対して関数$\Delta_{ij1}(x)$に従い移動量を計算する。移動量を画像化した帯域画素値は移動量画像の帯域画素値として扱われる。

【０１７１】ステップ１７０７で、AOU_{ij}が近傍の配列演算ユニット１００と通信することにより、重複情報画像１３２の各帯域画素値に対して関数$\Lambda_{ij1}(x)$に従い移動させる。移動した帯域画素値は新たに重複情報画像１３２の帯域画素値として扱われる。

【０１７２】ステップ１７０８で、ステップ１７０５からステップ１７０７までの繰り返し回数を表す移動回数が指定回数に達したかどうか判断する。もし移動回数が指定回数に達していなければ（ステップ１７０８：ＮＯ）、ステップ１７０５に戻る。もし移動回数が指定回数に達していれば（ステップ１７０８：ＹＥＳ）、ステップ１７０９に移行する。なおこの指定回数は形成エッジ情報画像１１５のサイズや形成エッジ情報１１４が表す物体のサイズ、さらには近傍のサイズｑにより決定される。利用目的に応じて適切なパラメータを設定すれば、指定回数を大目に決定しても問題はないが、あまり指定回数を多くしすぎると、位置及び大きさの検出に要する時間が長くなる。

【０１７３】ステップ１７０９で、AOU_{ij}が近傍の配列演算ユニット１００と通信することにより、重複情報画像１３２の各帯域画素値に対して関数$\Delta'_{ij1}(x)$に従い移動量を計算する。移動量を画像化した帯域画素値は移動量画像の帯域画素値として扱われる。

【０１７４】ステップ１７１０で、AOU_{ij}が近傍の配列演算ユニット１００と通信することにより、重複情報画像１３２の各帯域画素値に対して関数$\Lambda'_{ij1}(x)$に従い移動させる。移動した帯域画素値は新たに重複情報画像１３２の帯域画素値として扱われる。

【０１７５】ステップ１７１１で、重複情報画像１３２の帯域画素値を出力する。その後ステップ１７０３に戻る。

【０１７６】なお重複情報画像１３２の各重複情報１３１はその位置を中心とした周辺にある粗エッジ情報１１２の総数を表すので、結果的にその位置を中心とした物体の大きさを意味することになる。

【０１７７】これにより、配列演算ユニット１００から構成されるデータ処理装置１１０を用いて、位置／大きさ検出手段１７は粗エッジ情報画像１１３から重複情報画像１３２を生成することができる。

【０１７８】ここで図１６のアルゴリズムにおいて粗エッジ情報１１２から構成される粗エッジ情報画像１１３の代りに物体領域１４１を表す物体領域画像１４２を用いると、図１７に示すように、データ処理装置１１０によって実現される位置／大きさ検出手段１７は物体領域１４１を表す物体領域画像１４２からも重複情報１３１を表す重複情報画像１３２を生成することができる。た

だし物体領域画像１４２を用いた場合、重複情報画像１３２の各重複情報１３１はその位置を中心とした物体領域１４１の画素の総数を表すので、結果的にその位置を中心とした物体の面積を意味することになる。したがって重複情報画像１３２から物体の大きさを求める場合には重複情報１３１の平方根を取るなど注意を要する。

【０１７９】データ処理装置１１０によって実現される画像保持手段３８がデジタル画像１１１、粗エッジ情報画像１１３、形成エッジ情報画像１１５、重複情報画像１３２、物体領域画像１４２、正規化画像１４５、及びマッチング結果画像１４７、を保持及び出力するために、二次元格子状に配列された配列演算ユニット１００は同期して並列に動作する。格子上ｉ行ｊ列に配置された配列演算ユニット１００をAOU_{ij}とすると、デジタル画像１１１に対するAOU_{ij}のアルゴリズムは図１８のようになる。

【０１８０】ステップ３８０１で、AOU_{ij}を格子上のｉ行ｊ列に配置する。これは論理的であれ物理的であれ、AOU_{ij}の近傍を決定するために必要である。

【０１８１】ステップ３８０２で、AOU_{ij}の近傍や変数の初期値を設定する。

【０１８２】ステップ３８０３で、順次入力されるデジタル画像１１１が無くなったかどうか判断する。もしデジタル画像１１１が無ければ（ステップ３８０３：ＹＥＳ）、アルゴリズムを終了する。もしデジタル画像１１１があれば（ステップ３８０３：ＮＯ）、ステップ３８０４に移行する。ただし特定の画像サイズのみに対して配列演算ユニット１００を実装する場合には、無限ループにしても良い。

【０１８３】ステップ３８０４で、デジタル画像１１１のｉ行ｊ列の画素を帯域数分入力する。このためAOU_{ij}は少なくとも帯域数分の画像データを記憶するメモリ１０２を必要とする。

【０１８４】ステップ３８０５で、出力先の装置が必要とすればデジタル画像１１１のフォーマットを変換する。特にデジタル画像１１１の帯域数を１にしたり、前入力データから構成される入力画像の帯域数が４以上の場合にデジタル画像１１１の帯域数を３にして、アナログ信号を生成しやすくする場合に便利である。さもなくば何もしない。

【０１８５】ステップ３８０６で、処理速度の異なる出力先の装置に画像データを確実に送信できるように、デジタル画像１１１のｉ行ｊ列の画素を記憶する。

【０１８６】ステップ３８０７で、デジタル画像１１１の帯域画素値を出力する。その後ステップ３８０３に戻る。

【０１８７】これにより、配列演算ユニット１００から構成されるデータ処理装置１１０を用いて、画像保持手段３８がデジタル画像１１１を出力先の装置に出力することができる。

【０１８８】ここまではデータ処理装置１１０が１個又は２個の画像を入力して近傍処理のみからなる画像処理を行うような幾つかの手段について説明してきた。しかしながらパターンマッチング手段２９は非常に多数の画像を用いなければならない。そこで以下ではパターンマッチング手段２９で用いる近傍処理を示しながら、データ処理装置１１０によってパターンマッチング手段２９を実現する方法について説明する。

【０１８９】まず任意の画像を$\underline{x}$とし、ｎ個のテンプレート画像１４６を$\underline{y}^1$、$\underline{y}^2$、、、$\underline{y}^h$、、、$\underline{y}^n$とする。ただし、全てのテンプレート画像１４６の帯域数は画像$\underline{x}$の帯域数と同じである。自然数ｇを用いると、マッチング結果画像１４７のｉ行ｊ列のマッチング結果δ_{ij1}は、数式３７に従って画像$\underline{x}$及びテンプレート画像１４６のｉ行ｊ列の画素を比較し、画像ｘの画素に最も似ている画素を有するテンプレート画像１４６の番号を指し示す。なおマッチング結果画像１４７は単帯域画像となるので、便宜上帯域数１の画像として取り扱うことにする。したがってマッチング結果δ_{ij1}の第３添字は１となっている。

【０１９０】

【数３７】

$$\delta_{ij1}=\begin{cases} g & \text{if } \sum_k (x_{ijk}-y^g_{ijk})^2=\min_{1\le h\le n}\sum_k (x_{ijk}-y^h_{ijk})^2 \\ & \text{for } 1\le g\le n \text{ and only one of } g\ , \\ 0 & \text{otherwise.} \end{cases}$$

【０１９１】ここで数式３７に従って生成されたマッチング結果δ_{ij1}はマッチング結果画像１４７全体において必ずしも統一されていない。テンプレート画像１４６が多数ある場合、マッチング結果画像１４７はむしろモザイク状になる可能性が高い。そこでデータ処理装置１１０がマッチング結果δ_{ij1}とそのｑ近傍内のマッチング結果に対するヒストグラムを計算し、マッチング結果δ_{ij1}を収斂する方法を以下に示す。

【０１９２】任意の単帯域画像ｘがマッチング結果画像１４７である場合、自然数ｇ、実数ｕとｖを用いると、マッチング結果画像１４７は数式３８及び３９に従って更新される。なおマッチング結果画像１４７は単帯域画像となるので、便宜上帯域数１の画像として取り扱うことにする。したがって関数$\Psi_{ij1}(x)$の第３添字は１となっている。

【０１９３】

【数３８】

$$\Psi_{ij1}(\underline{x}) = \begin{cases} g & \text{if } \mathrm{eq}(g, x_{ij1}) + \sum_{p(l,m,1)\in P_{ij1}(q)} \mathrm{eq}(g, x_{lm1}) = \\ & \max_{1\le h\le n}\{\mathrm{eq}(h, x_{ij1}) + \sum_{p(l,m,1)\subset P_{ij1}(q)} \mathrm{eq}(h, x_{lm1})\} \\ & \text{and } 2\{\mathrm{eq}(g, x_{ij1}) + \sum_{p(l,m,1)\in P_{ij1}(q)} \mathrm{eq}(g, x_{lm1})\} \ge N_{ij1} \\ & \text{for } 1 \le g \le n \text{ and only one of } g\ , \\ 0 & \text{otherwise.} \end{cases}$$

【0194】
【数39】

$$\mathrm{eq}(u, v) = \begin{cases} 1 & \text{if } u = v, \\ 0 & \text{otherwise.} \end{cases}$$

【0195】マッチング結果画像147が変化しなくなるまでデータ処理装置110が数式38及び39を繰り返し計算することにより、マッチング結果画像147全体のマッチング結果を収斂することができる。このとき画像xとテンプレート画像146の組み合わせにより、マッチング結果は次のように収斂する。もし画像xの約半分の画素が特定のテンプレート画像146の画素に最も類似していれば、マッチング結果画像147の殆んどのマッチング結果はこの特定のテンプレート画像146の番号に収斂する。しかしながら画像xの幾つかの画素の塊が幾つかの異なるテンプレート画像146の画素の塊と類似していれば、マッチング結果画像147には0で囲まれた幾つかのテンプレート画像146の番号の塊ができる。さらに画像xがテンプレート画像146の集合と相関がなければ、マッチング結果画像147のマッチング結果は殆んど0となる。したがってデータ処理装置110によって実現されるパターンマッチング手段29は、画像xに最も似ているテンプレート画像146を特定することは難しいが、テンプレート画像146の中から幾つかの似ているテンプレート画像146を選択することができると考えられる。

【0196】なおタッチセンサにパターンマッチング手段29を実装した場合、タッチセンサに接続される後段の処理装置においてパターンマッチング手段29のマッチング結果画像147から移動物体及び静止物体の種別を生成する過程では、パターンマッチング手段29によって生成されたマッチング結果画像147が列挙するテンプレート画像146の類似候補の中から多数決などを行うことにより、最も有力な候補1つを選択するだけで良い。

【0197】図19に示すように、データ処理装置110によって実現されるパターンマッチングが、テンプレート画像146のうち正規化画像145に最も似ている画像の番号を示すマッチング結果から構成されるマッチング結果画像147を生成するために、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上i行j列に配置された配列演算ユニット100をAOU$_{ij}$とすると、AOU$_{ij}$のアルゴリズムは図20のようになる。

【0198】ステップ2901で、AOU$_{ij}$を格子上のi行j列に配置する。これは論理的であれ物理的であれ、AOU$_{ij}$の近傍を決定するために必要である。

【0199】ステップ2902で、AOU$_{ij}$の近傍や変数の初期値を設定する。近傍の設定においては、前記各関数で使う近傍サイズqを個別に決めても良いし、全部を統一しても良い。本発明のデータ処理装置110が生成したマッチング結果画像147の正確さを上げるためには近傍サイズqを全て大きな値に設定することが望ましい。しかしながらマッチング結果を更新するための計算時間の制約や、入力される正規化画像145のサイズなどにより、パターンマッチングは必要に応じて適宜近傍サイズを変えることで対処することができる。

【0200】ステップ2903で、順次入力されるテンプレート画像146が無くなったかどうか判断する。もしテンプレート画像146が無ければ（ステップ2903：YES）、ステップ2905に移行する。もしテンプレート画像146があれば（ステップ2903：NO）、ステップ2904に移行する。

【0201】ステップ2904で、テンプレート画像146のi行j列の画素を帯域数分入力する。このためAOU$_{ij}$は少なくとも帯域数とテンプレート画像146の数を掛けた分の画像データを記憶するメモリ102を必要とする。その後ステップ2903に戻る。

【0202】ステップ2905で、順次入力される正規化画像145が無くなったかどうか判断する。もし正規化画像145が無ければ（ステップ2905：YES）、アルゴリズムを終了する。もし正規化画像145があれば（ステップ2905：NO）、ステップ2906に移行する。ただし特定の画像サイズのみに対して配列演算ユニット100を実装する場合には、無限ループにしても良い。

【0203】ステップ2906で、正規化画像145のi行j列の画素を帯域数分入力する。このためAOU$_{ij}$は少なくとも帯域数分の画像データを記憶するメモリ102を必要とする。

【0204】ステップ2907で、正規化画像145とテンプレート画像146からマッチング結果画像147のマッチング結果δ_{ij1}を計算する。マッチング結果は正規化画像145に最も近いテンプレート画像146の番号を表す帯域画素値となる。

【0205】ステップ2908で、AOU$_{ij}$が近傍の配列演算ユニット100と通信することにより、マッチング結果画像147の各帯域画素値に対して関数Ψ

$_{ij1}$（$\underline{x}$）に従いマッチング結果を更新する。更新された帯域画素値は再びマッチング結果画像１４７の帯域画素値として扱われる。ここで関数Ψ_{ij1}（x）は本来マッチング結果画像１４７の帯域画素値が変化しなくなるまで繰り返される。ただし計算時間の制約、入力される正規化画像１４５の品質、更新されたマッチング結果画像１４７に求められる品質などにより、更新処理は適当な繰り返し回数で計算を打ち切った方が良い。

【０２０６】ステップ２９０９で、マッチング結果画像１４７の帯域画素値を出力する。その後ステップ２９０５に戻る。

【０２０７】これにより、配列演算ユニット１００から構成されるデータ処理装置１１０を用いて、パターンマッチング手段２９は正規化画像１４５からマッチング結果画像１４７を生成することができる。

【０２０８】ここまでは配列演算ユニット１００から構成されるデータ処理装置１１０を用いて、近傍処理のみからなる画像処理を行う方法について説明してきた。以下では配列演算ユニット１００から構成されるデータ処理装置１１０を用いて、近傍処理のみで物体／背景分離手段１６を行う方法について説明する。

【０２０９】まず非線形振動子は一般に引き込み現象を起こす。この引き込み現象とは、リミットサイクルやアトラクタなどのような周期的挙動において、異なる周期を持つ非線形振動子が相互作用して簡単な定数比の周期で振動するよう制約される現象である。このとき１つの非線形振動子の振動を変化させると他の非線形振動子の振動も合わせて変化するので、これらの非線形振動子は同期している。しかも非線形振動子の相互作用を調整することにより、お互いの振動の位相差を極力小さくさせたり大きくさせたりできる。そこでこの相互作用を操作すると、非線形振動子の一群を、異なる位相を持つ複数のグループに分割することができる。物体／背景分離手段１６はこのような非線形振動子の引き込み現象を利用して、エッジ情報画像中のエッジ情報を境界とするように物体と背景を分離して、物体領域１４１を表す物体領域画像１４２を生成する。なお、ここでは非線形振動子としてファン・デル・ポールを用いた場合を例に説明する。

【０２１０】まず、二次元格子状に配列した非線形振動子から構成される非線形振動子ネットワークにおいて、ｉ行ｊ列にある非線形振動子をω_{ij}とすると、非線形振動子ω_{ij}のｑ近傍にある非線形振動子の集合Ω_{ij}（ｑ）は数式４０によって表される。ただしｑは４、８、２４、４８、８０、１２０、$(2r+1)^2-1$と続く数列であり、ｒは自然数である。なおネットワークサイズをはみ出した非線形振動子が近傍集合Ω_{ij}（ｑ）に含まれる場合には、非線形振動子ω_{ij}を代用するものとする。これにより辺縁処理は自動的に行われる。したがって近傍集合Ω_{ij}（ｑ）の要素の数は常にｑとなる。なおこのことから判る通り、非線形振動子ネットワークは単帯域画像と同じ扱いになる。表現を簡単にするため、非線形振動子ネットワークでは添字は幅方向と高さ方向の２つのみを使う。

【０２１１】

【数４０】

$$\Omega_{ij}(q)=\begin{cases}\{\omega_{i+1,j},\ \omega_{i,j+1},\ \omega_{i-1,j},\ \omega_{i,j-1}\} & \text{if } q=4,\\ \{\omega_{l,m}\ |i-r\le l\le i+r,\ j-r\le m\le j+r,\ \omega_{l,m}\ne\omega_{i,j}\} & \text{if } q=(2r+1)^2-1.\end{cases}$$

【０２１２】次に、非線形振動子はq_a近傍に含まれる近傍集合Ω_{ij}（q_a）にある非線形振動子との間で数式４１に従い計算される結合値τ_{ijkl}によって結合される。なお対数表を用いない場合には数式４２による近似も可能である。またμ、νは適当な正の定数である。

【０２１３】

【数４１】

$$\tau_{ijkl}=\mu\,\mathrm{sinc}\left(\frac{(i-k)^2+(j-l)^2}{\nu^2}\right)\quad \text{for } \forall\omega_{kl}\in\Omega_{ij}(q_a)$$

【０２１４】

【数４２】

$$\mathrm{sinc}(x)\approx\begin{cases}1-2|x|^2+|x|^3 & \text{if } 0\le|x|<1,\\ 4-8|x|+5|x|^2-|x|^3 & \text{if } 1\le|x|<2,\\ 0 & \text{otherwise.}\end{cases}$$

【０２１５】非線形振動子ネットワークの全ての非線形振動子が完全に同位相で同期した場合、プロセッサ１０１で計算する限り、非線形振動子ω_{ij}は永久に同位相のまま動作し続けてしまう。そこで外乱ρ_{ij}を与えればこのような状態を回避することができる。外乱としては疑似乱数を用いることもできるが、数式４３のような簡単な式で求めても十分である。なおζ_{ij}はエッジ情報画像のｉ行ｊ列のエッジ情報の有無を表す。エッジ情

報があれば1とし、なければ0とする。またκは適当な正の定数である。

【0216】

【数43】

$$\rho_{ij} = \kappa\zeta_{ij}$$

【0217】非線形振動子ω_{ij}が近傍集合Ω_{ij}（q_a）の非線形振動子ω_{kl}と同期するために、数式44に従い近傍入力合計σ_{ij}を計算する。

【0218】

【数44】

$$\sigma_{ij} = \sum_{\omega_{kl}\in\Omega_{ij}(q_a)} \tau_{ijkl}(1-\zeta_{kl})\xi_{kl}(\psi_{kl}-\xi_{ij})$$

【0219】ファン・デル・ポール非線形振動子ω_{ij}を構成する2つのパラメータϕ_{ij}とψ_{ij}は数式45及び46に従って計算される。なおγ、εは適当な正の定数である。

【0220】

【数45】

$$\frac{d\phi_{ij}}{dt} = \psi_{ij}$$

【0221】

【数46】

$$\frac{d\psi_{ij}}{dt} = -\gamma\phi_{ij} - \epsilon(1-\phi_{ij}^2)\psi_{ij} + \sigma_{ij} + \rho_{ij}$$

【0222】非線形振動子を物体領域141と背景領域に分離するためには全ての非線形振動子の位相のずれを計算する必要があるが、単純に物体領域141と背景領域の2つに分離するだけであるので、パラメータψ_{ij}がしきい値θ以上か未満かで位相ずれを計算する。物体領域141と背景領域を分離した結果を出力する出力λ_{ij}は数式47によって求められる。なおθは適当な正の定数である。

【0223】

【数47】

$$\lambda_{ij} = \begin{cases} 1 & \text{if } \psi_{ij} \geq \theta\ , \\ 0 & \text{otherwise.} \end{cases}$$

【0224】エッジ情報が物体と背景を分離するのに不十分であった場合にはエッジ情報を補間しなければならない。そのために非線形振動子ω_{ij}のq_b近傍にある非線形振動子の集合Ω_{ij}（q_b）中でいくつの非線形振動子が位相ずれを起こしているか求める必要がある。そこで数式48によって輪郭パラメータη_{ij}を計算する。

【0225】

【数48】

$$\eta_{ij} = \sum_{\omega_{kl}\in\Omega_{ij}(q_b)} \lambda_{ij}\lambda_{kl} + (\lambda_{ij})^2$$

【0226】この結果を基にエッジ情報の補間割合を示す境界パラメータξ_{ij}を数式49によって計算する。なおα、β、η_{min}、η_{max}は適当な正の定数である。

【0227】

【数49】

$$\frac{d\xi_{ij}}{dt} = \begin{cases} -\alpha\xi_{ij} & \text{if } \eta_{min} \leq \eta_{ij} \leq \eta_{max}\ , \\ \beta(1-\xi_{ij}) & \text{otherwise.} \end{cases}$$

【0228】ここでは非線形振動子としてファン・デル・ポールの場合を説明したが、この他にブラッセレータのようなリミットサイクルで安定する非線形振動子や、ローレンツアトラクタやレスラー方程式のアトラクタを発生するカオス振動子など、引き込み現象を起こす任意の非線形振動子でも動作可能である。その場合はパラメータϕ_{ij}とψ_{ij}を各々の非線形振動子のパラメータで置き換えるなり追加すれば良い。その際に適当なパラメータに近傍入力合計σ_{ij}と外乱ρ_{ij}を加えるだけである。但し、カオス振動子の場合には特に外乱ρ_{ij}を必要としない。

【0229】数式40から数式49までを用いることにより、物体／背景分離手段16を実装することができるデータ処理装置110の全ての配列演算ユニット100のアルゴリズムを記述することができる。以下では、データ処理装置110中の任意の配列演算ユニット100のアルゴリズムを用いて、物体／背景分離手段16について説明する。

【0230】図21に示すように、データ処理装置110によって実現される物体／背景分離手段16が、三角形のエッジ情報151を用いて三角形の内側領域152と三角形の外側領域153に分離するために、二次元格子状に配列された配列演算ユニット100は同期して並列に動作する。格子上i行j列に配置された配列演算ユニット100をAOU_{ij}とすると、AOU_{ij}のアルゴリズムは図22のようになる。

【0231】ステップ1601で、AOU_{ij}を格子上のi行j列に配置する。

【0232】ステップ1602で、数式41及び42に基づいて近傍同士ω_{ij}とω_{kl}を結合値τ_{ijkl}で接続する。

【0233】ステップ1603で、非線形振動子のパラメータϕ_{ij}とψ_{ij}に適当な初期値を設定する。

【0234】ステップ1604で、順次入力される形成エッジ情報画像115が無くなったかどうか判断する。もし形成エッジ情報画像115が無ければ（ステップ1604：YES）、アルゴリズムを終了する。もし形成エッジ情報画像115があれば（ステップ1604：NO）、ステップ1605に移行する。ただし特定の帯域数及び画像サイズのみに対して配列演算ユニット100を実装する場合には、無限ループにしても良い。

【0235】ステップ1605で、形成エッジ情報114のξ_{ij}を入力する。

【0236】ステップ1606で、直前に入力した形成エッジ情報114のζ_{ij}から数式43に従って外乱ρ_{ij}を計算する。

【0237】ステップ1607で、近傍集合Ω_{ij}(q_a)中の非線形振動子ω_{kl}がある配列演算ユニット100のAOU_{kl}からζ_{kl}、ξ_{kl}、ψ_{kl}を入力して、合計値σ_{ij}を数式44に従って計算する。

【0238】ステップ1608で、非線形振動子のパラメータϕ_{ij}、ψ_{ij}を数式45及び46に従って計算する。即ち、これらの数式に示す微分方程式をルンゲ・クッタ法で解く。

【0239】ステップ1609で、非線形振動子の出力λ_{ij}を数式47に従って計算する。ここで、$\psi_{ij} \geqq \theta$であれば$\lambda_{ij}=1$とし、それ以外であれば$\lambda_{ij}=0$とする。

【0240】ステップ1610で、近傍集合Ω_{ij}(q_b)中の非線形振動子ω_{kl}がある配列演算ユニット100のAOU_{kl}からλ_{kl}を入力して、輪郭パラメータη_{ij}を数式48に従って計算する。

【0241】ステップ1611で、境界パラメータξ_{ij}を数式49に従って計算する。即ち、この数式に示す微分方程式を差分法若しくはルンゲ・クッタ法で解く。

【0242】ステップ1612で、ステップ1606からステップ1611までの繰り返し回数を表す分離回数が指定回数に達したかどうか判断する。もし分離回数が指定回数に達していなければ(ステップ1612:NO)、ステップ1606に戻る。もし分離回数が指定回数に達していれば(ステップ1612:YES)、ステップ1613に移行する。

【0243】ステップ1613で、物体領域画像142の帯域画素値となる非線形振動子の出力λ_{ij}を出力する。その後ステップ1604に戻る。

【0244】なおステップ1612での分離回数を求めるには、次のような方法を用いることができる。物体/背景分離手段16では、画像サイズが一定であれば非線形振動子の初期状態に関わらずおおよそ全ての形成エッジ情報114においてある一定時間で分離が終了するので、事前にこの時間を計っておいてステップ1606からステップ1611までの繰り返し回数を求めておけば良い。これは非線形振動子の初期状態が一定の範囲内にあれば、引き込み現象により非線形振動子が同期するまでの時間はあまり大差がないからである。

【0245】このように非線形振動子を計算するだけで、三角形のエッジ情報151を用いて三角形の内側領域152と三角形の外側領域153を分離することができるのは、非線形振動子の性質である引き込み現象を利用しているからである。つまり、2つの非線形振動子を正の結合値で結合した場合は同位相になろうとし、負の結合値で結合した場合は位相差が極力大きくなろうとする。この性質を用いると、二次元格子状に並んだ非線形振動子を近傍同士正の結合値で結合することで、直接結合していない非線形振動子同士が同位相になる。さらに形成エッジ情報114を挟む画素の場所にある非線形振動子同士を負の結合値で結合すると、エッジ情報の両側がお互いに位相を極力ずらし合う。このようにすることで、全ての非線形振動子を結合することもなく三角形のエッジ情報151の内側と外側とで各々異なる位相集合ができる。したがって物体/背景分離手段16は図21のような三角形の内側領域152と三角形の外側領域153に分離する。このとき三角形の内側領域152と三角形の外側領域153の位相差は90度を越えて可能な限り180度に近づき、三角形と背景領域が分離できる。

【0246】ここで重要なことは、本実施形態では、形成エッジ情報114が得られる度に次に示すような方法で結合値を擬似的に変更していることである。まず数式41及び42で定めたように、非線形振動子ω_{kl}を非線形振動子ω_{ij}に結合するための結合値をτ_{ijkl}とする(ステップ1602参照)。形成エッジ情報114のうちζ_{ij}とζ_{kl}は共に、エッジがある場合には1、ない場合には0である。形成エッジ情報114のうちζ_{ij}とζ_{kl}を入力したら(ステップ1605参照)、配列演算ユニット100のAOU_{kl}からAOU_{ij}に形成エッジ情報114ζ_{kl}が転送され、AOU_{ij}では結合値τ_{ijkl}($1-\zeta_{kl}$)を計算して結合値τ_{ijkl}の代用とする(ステップ1607参照)。この代用された結合値τ_{ijkl}($1-\zeta_{kl}$)に対して境界パラメータξ_{ij}が0から1までの倍率として作用する(ステップ1607参照)。

【0247】図23に示す通り、形成エッジ情報114が破線状態の三角形のエッジ情報154となった場合には破線の補間をする必要がある。まず初めに破線状態の三角形のエッジ情報154を用いてシステムを動作させる(ステップ1605参照)と、破線状態の三角形のエッジ情報154の内側と外側で位相差がおよそ90度を越えるようになるが、三角形の内側と外側の境界部分は不明確である。そこで各AOU_{ij}が非線形振動子の出力λ_{ij}を計算する(ステップ1609参照)。この出力λ_{ij}が1の場合、近傍の非線形振動子のうちλ_{kl}が1である非線形振動子をω_{kl}とすると、パラメータψ_{ij}とψ_{kl}が共にθ以上となる。つまりλ_{ij}とλ_{kl}はおよそ同位相であり、θが正値であれば最悪でも位相差が90度を越えることはない。この位相差の最大値はθの値によって決まり、λ_{ij}とλ_{kl}が共に1となる範囲でθを大きくしていくと、この位相差は0度に近づいていく。したがってλ_{ij}とλ_{kl}と用いると、近傍の非線形振動子うちおよそ同位相であるものの数を表す輪郭パラメータη_{ij}は数式48に従って計算される(ステップ1610参照)。続いてこの輪郭パラメー

タη_{ij}が近傍全体のうち、およそ半分であれば結合値の倍率である境界パラメータξ_{ij}を数式４９に従って減少させ、それ以外であれば数式４９に従って増加させる（ステップ１６１１参照）。例えば、８近傍の場合は３から５の間であれば数式４９に従って境界パラメータを減少させるとよい。この過程を繰り返し動作させ続けると、図２３に示す破線状態の三角形のエッジ情報１５４が与えられた場合、破線三角形の内側領域１５５と破線三角形の外側領域１５６に分離される。

【０２４８】図２４に示す通り、２つの三角形が重なりあっている場合は、前方の三角形のエッジ情報１５７と後方の三角形のエッジ情報１５８が得られる。このとき前方三角形の内側領域１５９と後方三角形の内側領域１６０と二重三角形の背景領域１６１の３つの領域の非線形振動子の位相がお互いにずれることにより、３つの領域に分離される。また図２５に示す通り、２つの重なった円形のエッジ情報１６２が破線であっても、前方円形の内側領域１６３と後方円形の内側領域１６４と二重円の背景領域１６５の３つに分離される。

【０２４９】これにより、配列演算ユニット１００から構成されるデータ処理装置１１０を用いて、物体／背景分離手段１６は形成エッジ情報画像１１５の形成エッジ情報１１４を境界として物体領域１４１と背景を分離することができる。

【０２５０】そこでデジタル技術を用いて画像記憶手段１２、画像振動手段、エッジ情報生成手段１４、エッジ情報形成手段１５、物体／背景分離手段１６、領域正規化手段２７、位置／大きさ検出手段１７、パターンマッチング手段２９、及び画像保持手段３８、を実装するために、配列演算ユニット１００はデータ処理装置１１０中で図７のように二次元格子状に配列され、さらに配列演算ユニット１００はデータ処理装置１１０中の隣接する配列演算ユニット１００だけと相互に通信できるように配線される。つまり４近傍同士が直接配線されることになる。これにより８近傍同士を配線する場合に比べて、少ない電子部品と配線量で、同程度に高速に動作し、しかも将来近傍サイズを拡張する場合にも簡単に拡張性を有することができる。

【０２５１】配列演算ユニット１００は図２６に示す通り、画像処理における数式を計算するためのプロセッサ（PROCESSOR）１０１と、数式で使われる全てのパラメータ、定数、関数及びオペレータを記憶するためのメモリ（MEMORY）１０２と、近傍の配列演算ユニット１００と通信するためのコントローラ（CONTROLER）１０３から構成され、プロセッサ１０１及びコントローラ１０３はクロック信号（CLOCK）を入力し、プロセッサ１０１はメモリ１０２及びコントローラ１０３に対して読み込み（READ）及び書き込み（WRITE）を送信することにより通信を制御する。プロセッサ１０１はアドレスバス５１で指定したアドレス（ADDRESS）によりメモリ１０２及びコントローラ１０３の任意のメモリ素子及びレジスタを選択することができる。またプロセッサ１０１はデータバス５２を介してメモリ１０２及びコントローラ１０３と双方向に通信可能に接続され、アドレスバス５１で指定された任意のメモリ素子及びレジスタのデータ（DATA）にアクセスすることができる。配列演算ユニット１００が１つ以上の入力画素から構成される前入力データ群（FRONT INPUT DATA SET）を入力すると、コントローラ１０３は前入力データ群をメモリ１０２に記憶させる。またコントローラ１０３は、関数により作成されたメモリ１０２中の計算データを隣接する配列演算ユニット１００に送信すると共に、隣接する配列演算ユニット１００から受信した計算データをメモリ１０２に記憶させ、さらに必要ならば、入力した以外の配列演算ユニット１００に転送する。最終的にコントローラ１０３は、出力画像の画像データを結果データ（RESULT DATA）として出力する。

【０２５２】このように各配列演算ユニット１００にコントローラ１０３を搭載する理由は、配列演算ユニット１００同士が通信している間にプロセッサ１０１が動作できるので、プロセッサ１０１は通信による待ち時間中にも計算することができて高速処理が実現できるからと、近傍の配列演算ユニット１００の数を変化させてもハードウェアを変更する必要がないからと、コントローラ１０３が画像の辺縁処理、つまり画像中の縁の画素に対する例外処理を自動的に行えるので、プロセッサ１０１のプログラムは辺縁処理をする必要がなくなり極めて単純になるからと、である。

【０２５３】プロセッサ１０１とメモリ１０２は汎用的なデジタル回路を用いることができる。コントローラ１０３の具体的な回路図は図２７に示す通りである。アドレスバッファ（ADDRESS BUFFER）５３はアドレスバス（ADDRESS BUS）５１を介してプロセッサ１０１からアドレス（ADDRESS）を受取り、アドレスデコーダ（ADDRESS DECODER）５４によって各レジスタ及びその他の電子回路ブロックを選択する。データバッファ（DATA BUFFER）５５はデータバス（DATA BUS）５２を介してプロセッサ１０１からデータ（DATA）を受取り、アドレスデコーダ５４で選択されたレジスタと内部データバス５６を介して排他的に通信する。通信方向は読み込み（READ）によって指定される。コントローラ１０３からプロセッサ１０１に向けてデータが送信される際、読み込みはアクティブになる。プロセッサ１０１からコントローラ１０３に向けてデータが送信される際、書き込み（WRITE）はアクティブになる。アドレスがフラグレジスタ（FLAG REGISTER）５７を指定した場合、データはフラグレジスタ５７に記憶され、フラグデコーダ（FLAG DECODER）５８によってデコードされ、クロック信号（CLOCK）を用いて、複数信号（SIGNALS）として隣接する配列演算ユニット１００に送信される。複数信号はフラグエン

コーダ（FLAG ENCODER）５９によって受信され、解析された後にステータスレジスタ（STATUS REGISTER）６０に記憶され、また受領（RECEIVE）として送信元の配列演算ユニット１００に返送される。受領は複数信号の送信元のフラグエンコーダ５９で受信され、結果として複数信号の送信完了が確認される。アドレスによってステータスレジスタ６０が選択されると、ステータスレジスタ６０の内容がデータバス５２を介してデータとしてプロセッサ１０１に送信される。１つ以上の入力画像（INPUT IMAGE）に対応した１つ以上の前入力送達（FRONT INPUT SEND）をフラグエンコーダ５９が受信すると１つ以上の入力画像からなる前入力データ群（FRONT INPUT DATA SET）が必要な記憶容量分用意された前入力データレジスタ（FRONT INPUT DATA REGISTER）６１に読み込まれる。アドレスによって前入力データレジスタ６１が選択されると、前入力データレジスタ６１の内容がデータとしてプロセッサ１０１に送信される。プロセッサ１０１が計算を完了したら、アドレスによって結果データレジスタ（RESULT DATA REGISTER）６２が選択され、結果データレジスタ６２が出力画像の画像データを結果データ（RESULT DATA）として読み込む。これと同時に、フラグエンコーダ５９が結果送達（RESULT SEND）を送信する。

【０２５４】近傍の配列演算ユニット１００から計算に必要なデータを求められたら、アドレスとして出力データレジスタ（OUTPUT DATA REGISTER）６３を選択し、近傍の配列演算ユニット１００に送信すべきデータを計算データ（CALCURATION DATA）として出力データレジスタ６３に読み込む。その後、隣接する全ての配列演算ユニット１００に計算データとして送信される。上側の配列演算ユニット１００から複数信号（SIGNALS）を受信したら計算データを上入力データレジスタ（UPPERINPUT DATA REGISTER）６４に読み込む。その後、アドレスにより上入力データレジスタ６４が選択されたら、上入力データレジスタ６４の内容が計算データとして送信される。下側、左側、右側の配列演算ユニット１００から複数信号を受信した場合も同様であり、下入力データレジスタ６５、左入力データレジスタ６６、右入力データレジスタ６７が同様に動作する。

【０２５５】各種バッファ、各種レジスタ、アドレスデコーダ５４の各ブロックは汎用的な電子回路である。フラグデコーダ５８とフラグエンコーダ５９は具体的には図２８と図２９に示すような入出力信号を有する。種別（TYPE）は出力データレジスタ（OUTPUT DATA REGISTER）６３に読み込まれた内容の種類を５ビットで表す。このビット数は配列演算ユニット１００が送受信すべき全ての計算データを区別するのに十分な値である。カウント－Ｘ（COUNT-X）及びカウント－Ｙ（COUNT-Y）は各々４ビットの符号なし整数を表し、配列演算ユニット１００の間の転送回数を示す。配列演算ユニット１００が計算データを送信する場合には各々のカウントが０となり、左右の配列演算ユニット１００から送信された計算データを再度送信する場合にはフラグエンコーダ５９のカウント－Ｘに１を足した値となり、上下の配列演算ユニット１００から送信された計算データを再度送信する場合にはフラグエンコーダ５９のカウント－Ｙに１を足した値となる。プロセッサ１０１が上下左右のうちどの方向に出力データレジスタ６３の内容を送信するかをフラグレジスタ５７の送達フラグ（SEND FLAG）に指定した後で、出力データレジスタ６３を指定するアドレスデコーダ５４の中央デコーディング（CENTRAL DECODING）をフラグデコーダ５８が受信すると、フラグデコーダ５８が送達（SEND）を送達フラグの指定方向に合わせて出力する。送達フラグは４ビットで表し、配列演算ユニット１００の計算データを四方の配列演算ユニット１００に送信する場合にはプロセッサ１０１が１１１１と設定し、右側の配列演算ユニット１００から送信された計算データを上下左側に転送する場合はプロセッサ１０１が１１１０と設定し、左側から上下右側に転送する場合は１１０１と設定し、下側から上側に転送する場合は１０００と設定し、上側から下側に転送する場合は０１００と設定する。これにより、転送に重複がなくなり効率的に転送できるだけでなく、転送方向の決定規則が明確になっているので、種別、カウント－Ｘ及びカウント－Ｙを組み合わせることにより、フラグエンコーダ５９はどの配列演算ユニット１００からどの種別の計算データが送信されたかを判定することができる。結果データレジスタ６２に計算データが結果データとして読み込まれると同時にフラグデコーダ５８は、結果デコーディング（RESULT DECODING）を受信し、結果送達（RESULT SEND）を送信する。

【０２５６】フラグエンコーダ５９は四方のうちいずれかでも送達を受信したら、受信方向の種別とカウント－Ｘ、カウント－Ｙを受信し、その部分のステータスレジスタ６０の内容を更新する。この更新と同時に受信方向に受領を１にして送信する。送信元の配列演算ユニット１００のフラグエンコーダ５９では受領が１になった瞬間に受信し、ステータスレジスタ６０の受領ステータス（RECEIVE STATUS）を更新する。これにより各配列演算ユニット１００ではプロセッサ１０１がステータスレジスタ６０の受領ステータスを検査するだけで、どの入力データレジスタに有効な計算データが記憶されているか判断することができる。そこで例えば上入力データレジスタ６４に計算データが読み込まれていれば、プロセッサ１０１がアドレスを指定することにより上入力データレジスタ６４からデータを読み込むことができるが、同時にアドレスデコーダ５４から上デコーディング（UPPERDECODING）がフラグエンコーダ５９に送信され、受領ステータスのうち上部分が０に戻され、上側に向いた受領が０として送信される。下左右側の場合も同様に動作

する。フラグエンコーダ５９が１つでも入力画像用の前入力送達を受信したら、ステータスレジスタ６０のうち受信した前入力送達に対応する入力画像用の前入力送達ステータス（FRONT INPUT SEND STATUS）を１にする。またプロセッサ１０１が入力画像用の前入力データレジスタ６１からデータを読み込むとき、アドレスデコーダ５４がフラグエンコーダ５９に前デコーディング（FRONT DECODING）を送信し、受信した前入力送達に対応する前入力送達ステータスを０にする。プロセッサ１０１はステータスレジスタ６０の内容を読み込むことにより、前入力データレジスタ６１に最新の入力画像が記憶されているかどうか判断することができる。

【０２５７】プロセッサ１０１がコントローラ１０３を介して四方の配列演算ユニット１００に計算データを送信する場合のアルゴリズムを図３０に示す。図３０は、プロセッサ１０１によるプログラム制御と、フラグデコーダ５８及びフラグエンコーダ５９によるハードウェアロジックとの混成による処理を示すものである。図３０に対して、ステップ７１では、プロセッサ１０１がステータスレジスタ６０の内容を読み込む。ステップ７２では、読み込んだ内容のうち受領ステータスが全て０であるか否かを判断する。ＮＯなら処理を終了する。ＹＥＳならステップ７３に移行する。ステップ７３では、プロセッサ１０１が隣接する配列演算ユニット１００に送信するデータの種別とカウンタと送信方向を決定し、その内容をフラグレジスタ５７に書き込む。ステップ７４では、プロセッサ１０１が隣接する配列演算ユニット１００に送信するデータを出力データレジスタ６３に書き込む。ステップ７５では、出力データレジスタ６３の内容を計算データとして、隣接する配列演算ユニット１００に送信する。ステップ７６では、フラグレジスタ５７の送達フラグで指定された方向にのみ送達を１にして送信する。これによりプロセッサ１０１の１回の送信アルゴリズムは終了する。プロセッサ１０１は、送信すべきデータがメモリ１０２内で更新される度にこの送信アルゴリズムを開始する。

【０２５８】コントローラ１０３が上側の配列演算ユニット１００から計算データを受信する場合のアルゴリズムを図３１に示す。図３１は、フラグデコーダ５８及びフラグエンコーダ５９によるハードウェアロジックによる処理を示すものである。図３１に対して、ステップ８１では、フラグエンコーダ５９が送達を入力する。ステップ８２では、送達が１であるか否かをフラグエンコーダ５９が判断する。ＮＯなら処理を終了する。ＹＥＳならステップ８３に移行する。ステップ８３では、上入力データレジスタ６４が上側から送信された計算データを読み込む。ステップ８４では、フラグエンコーダ５９がステータスレジスタ６０のうち上側用の受領ステータスを１にすると同時に受領を１にして上側の配列演算ユニット１００に送信する。下左右側の場合も同様である。これによりコントローラ１０３の１回の受信アルゴリズムは終了する。コントローラ１０３は常時上下左右の配列演算ユニット１００からの送達を監視し、この送達を受信する度にこの受信アルゴリズムを開始する。

【０２５９】プロセッサ１０１が上入力データレジスタ６４からデータを受信する場合のアルゴリズムを図３２に示す。図３２は、プロセッサ１０１によるプログラム制御と、フラグデコーダ５８及びフラグエンコーダ５９によるハードウェアロジックとの混成による処理を示すものである。図３２に対して、ステップ９１では、プロセッサ１０１がステータスレジスタ６０の内容を読み込む。ステップ９２では、読み込んだ内容のうち上側用の受領ステータスが１であるか否かを判断する。ＮＯなら処理を終了する。ＹＥＳならステップ９３に移行する。ステップ９３では、プロセッサ１０１が上入力データレジスタ６４からデータを読み込む。ステップ９４では、フラグエンコーダ５９がステータスレジスタ６０のうち上側用の受領ステータスを０にすると同時に受領を０にして上側の配列演算ユニット１００に送信する。下左右側の場合も同様である。これによりプロセッサ１０１の１回の受信アルゴリズムは終了する。プロセッサ１０１は一定間隔でステータスレジスタ６０の内容を監視し、上下左右いずれかの受領ステータスが１である度にこの受信アルゴリズムを開始する。なおここではプロセッサ１０１が一定間隔でステータスレジスタ６０の内容を常時監視する場合を示したが、コントローラ１０３に割り込み回路を付加することにより、プロセッサ１０１は割り込みによりこの受信アルゴリズムを開始することもできる。

【０２６０】なおこの配列演算ユニット１００は、主に１つ以上の入力画像から１つの出力画像を生成することを前提に記述したが、用途に応じては計算途中の計算データを出力できるように回路を変更する必要がある。その際には、出力すべき計算データの数だけフラグデコーダ５８の結果送達を増やし、結果データレジスタ６２に読み込まれた計算データに対応する結果送達のみを１にするようにプログラムを変更するだけで良い。

【０２６１】以上、本実施形態を説明したが、本発明は上述の実施形態には限定されることはなく、当業者であれば種々なる態様を実施可能であり、本発明の技術的思想を逸脱しない範囲において本発明の構成を適宜改変できることは当然であり、このような改変も、本発明の技術的範囲に属するものである。

【０２６２】

【発明の効果】請求項１及び２記載の発明によれば、大規模集積回路（ＬＳＩ）及び圧力センサ２の製造メーカーは、この製造メーカーが保有するＬＳＩ及び圧力センサ製造技術をそのまま用いて高機能タッチセンサを製造することができる。したがって製造メーカーは短期間にしかも安価に高機能タッチセンサを製造できるので、本

発明の利用者は、家屋、コンビニエンスストア、スーパー、事務所、工場、一般道路、駐車場、駅、公益機関、車両、船舶、及び宇宙ステーションにおいて、入力装置、保持装置、検査装置、保安装置、警戒装置、計数装置、ペットロボット、受付ロボット、作業ロボット、及び福祉ロボットなど色々な用途に高機能タッチセンサを利用できるようになる。特に圧力センサ2が圧力の微妙な変化を捉えることができれば、本発明は筆圧、指圧及び指紋の凹凸を分類することができ、利用者は本発明を個人識別にも利用することができる。

【0263】請求項3記載の発明によれば、LSI及び圧力センサ2の設計者は、1個の機能コラム1を設計して二次元格子状に配列するだけで、適当なサイズの画像を射影することができる高機能タッチセンサを容易に設計することができる。したがってLSI及び圧力センサ2の設計者は、短期間でしかも安価に高機能タッチセンサを設計することができる。

【0264】請求項4記載の発明によれば、大規模集積回路（LSI）及び温度センサの製造メーカーは、この製造メーカーが保有するLSI及び温度センサ製造技術をそのまま用いて高機能タッチセンサを製造することができる。したがって製造メーカーは短期間にしかも安価に高機能タッチセンサを製造できるので、本発明の利用者は、家屋、コンビニエンスストア、スーパー、事務所、工場、一般道路、駐車場、駅、公益機関、車両、船舶、及び宇宙ステーションにおいて、入力装置、保持装置、検査装置、保安装置、警戒装置、計数装置、ペットロボット、受付ロボット、作業ロボット、及び福祉ロボットなど色々な用途に高機能タッチセンサを利用できるようになる。特に温度センサが温度の微妙な変化を捉えることができれば、本発明は体温及び指の汗の分布を分類することができ、利用者は本発明を個人識別にも利用することができる。

【0265】請求項5記載の発明によれば、大規模集積回路（LSI）及び電極の製造メーカーは、この製造メーカーが保有するLSI及び電極製造技術をそのまま用いて高機能タッチセンサを製造することができる。したがって製造メーカーは短期間にしかも安価に高機能タッチセンサを製造できるので、本発明の利用者は、家屋、コンビニエンスストア、スーパー、事務所、工場、一般道路、駐車場、駅、公益機関、車両、船舶、及び宇宙ステーションにおいて、入力装置、保持装置、検査装置、保安装置、警戒装置、計数装置、ペットロボット、受付ロボット、作業ロボット、及び福祉ロボットなど色々な用途に高機能タッチセンサを利用できるようになる。特に電極が水分の微妙な変化を捉えることができれば、本発明は指の表面抵抗、誘電率及び汗の分布を分類することができ、利用者は本発明を個人識別にも利用することができる。

【図面の簡単な説明】

【図1】1個の圧力センサ、1個のA／D変換回路、及び1個の配列演算ユニットを備えた機能コラムのブロック図である。

【図2】1個の圧力センサ、1個のA／D変換回路、1個の配列演算ユニット、及び1個の発振回路を備えた機能コラムのブロック図である。

【図3】圧力の位置及び大きさを検知する装置用の機能コラムのブロック図である。

【図4】発振回路を搭載し、圧力の変化を抽出して圧力の位置及び大きさを検知する装置用の機能コラムのブロック図である。

【図5】二次元格子状に機能コラムを配列した場合のタッチセンサのブロック図である。

【図6】機能コラムを大規模集積回路の実装面に対して垂直に実装した場合のタッチセンサのブロック図である。

【図7】二次元格子状に配列された配列演算ユニットのブロック図である。

【図8】本実施形態の画像記憶手段のアルゴリズムを示すフローチャートである。

【図9】本実施形態の画像振動手段のアルゴリズムを示すフローチャートである。

【図10】本実施形態のエッジ情報生成手段のアルゴリズムを示すフローチャートである。

【図11】デジタル画像を用いて粗エッジ情報を形成エッジ情報に形成する場合の説明図である。

【図12】本実施形態のエッジ情報形成手段のアルゴリズムを示すフローチャートである。

【図13】デジタル画像の切出領域を正規化する場合の説明図である。

【図14】本実施形態の領域正規化手段のアルゴリズムを示すフローチャートである。

【図15】エッジ情報画像中の物体の位置及び大きさを検出する場合の説明図である。

【図16】本実施形態の位置／大きさ検出手段のアルゴリズムを示すフローチャートである。

【図17】物体領域画像中の物体の位置及び大きさを検出する場合の説明図である。

【図18】本実施形態のデジタル画像用の画像保持手段のアルゴリズムを示すフローチャートである。

【図19】正規化画像に対してテンプレート画像の中からパターンマッチングをする場合の説明図である。

【図20】本実施形態のパターンマッチング手段のアルゴリズムを示すフローチャートである。

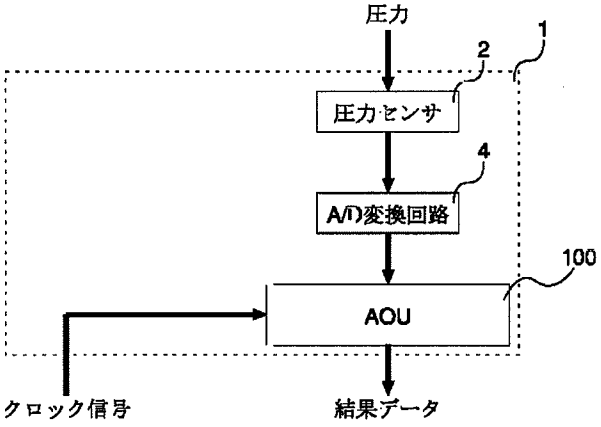
【図21】三角形のエッジ情報が三角形の内側領域と外側領域に分離する状態を示す説明図である。

【図22】本実施形態の物体／背景分離手段のアルゴリズムを示すフローチャートである。

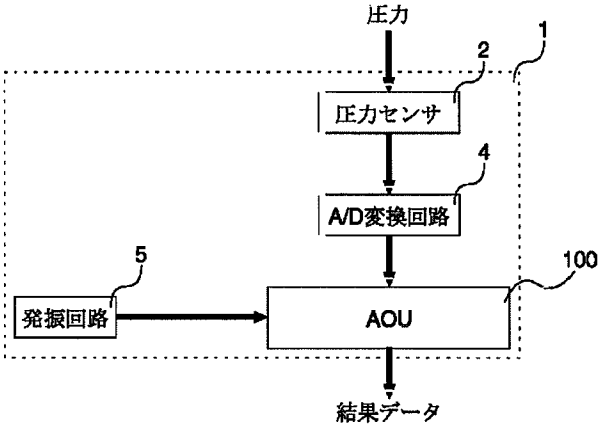
【図23】破線状態の三角形のエッジ情報が破線三角形の内側領域と外側領域に分離する状態を示す説明図であ

る。
【図２４】三角形を２つ重ねたエッジ情報が２つの三角形領域と背景領域に分離する状態を示す説明図である。
【図２５】２つの円形物体領域を重ねた時の破線状態のエッジ情報が２つの円形領域と背景領域に分離した状態を示す説明図である。
【図２６】配列演算ユニットの内部構造のブロック図である。
【図２７】コントローラのブロック図である。
【図２８】フラグデコーダの入出力信号を示す説明図である。
【図２９】フラグエンコーダの入出力信号を示す説明図である。
【図３０】プロセッサがコントローラを介して隣接する配列演算ユニットにデータを送信するアルゴリズムを示すフローチャートである。
【図３１】コントローラが隣接する配列演算ユニットからデータを受信するアルゴリズムを示すフローチャートである。
【図３２】プロセッサが上入力レジスタからデータを受信するアルゴリズムを示すフローチャートである。
【符号の説明】
１　機能コラム
２　圧力センサ
４　Ａ／Ｄ変換回路
５　発振回路
１２　画像記憶手段
１４　エッジ情報生成手段
１５　エッジ情報形成手段
１６　物体／背景分離手段
１７　位置／大きさ検出手段
２７　領域正規化手段
２９　パターンマッチング手段
３８　画像保持手段
５１　アドレスバス
５２　データバス
５３　アドレスバッファ
５４　アドレスデコーダ
５５　データバッファ
５６　内部データバス
５７　フラグレジスタ
５８　フラグデコーダ
５９　フラグエンコーダ
６０　ステータスレジスタ
６１　前入力データレジスタ
６２　結果データレジスタ
６３　出力データレジスタ
６４　上入力データレジスタ
６５　下入力データレジスタ
６６　左入力データレジスタ
６７　右入力データレジスタ
１００　配列演算ユニット
１０１　プロセッサ
１０２　メモリ
１０３　コントローラ
１１０　データ処理装置
１１１　デジタル画像
１１２　粗エッジ情報
１１３　粗エッジ情報画像
１１４　形成エッジ情報
１１５　形成エッジ情報画像
１３１　重複情報
１３２　重複情報画像
１４１　物体領域
１４２　物体領域画像
１４３　分離物体領域
１４４　正規化領域
１４５　正規化画像
１４６　テンプレート画像
１４７　マッチング結果画像
１５１　三角形のエッジ情報
１５２　三角形の内側領域
１５３　三角形の外側領域
１５４　破線状態の三角形のエッジ情報
１５５　破線三角形の内側領域
１５６　破線三角形の外側領域
１５７　前方の三角形のエッジ情報
１５８　後方の三角形のエッジ情報
１５９　前方三角形の内側領域
１６０　後方三角形の内側領域
１６１　二重三角形の背景領域
１６２　２つの重なった円形のエッジ情報
１６３　前方円形の内側領域
１６４　後方円形の内側領域
１６５　二重円の背景領域

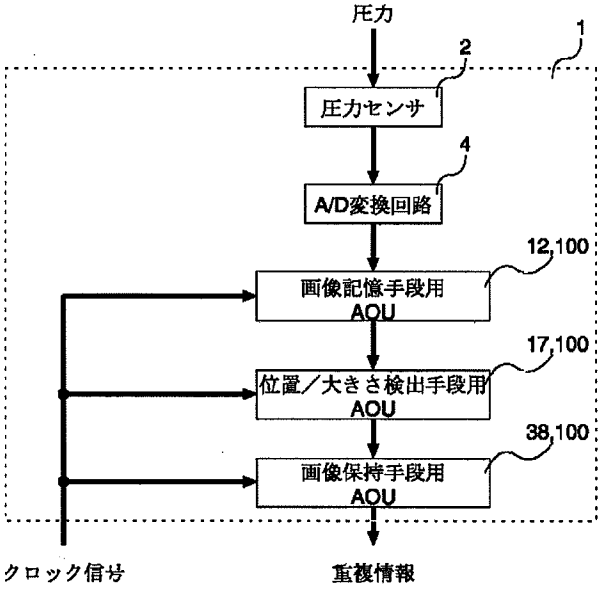
【図1】



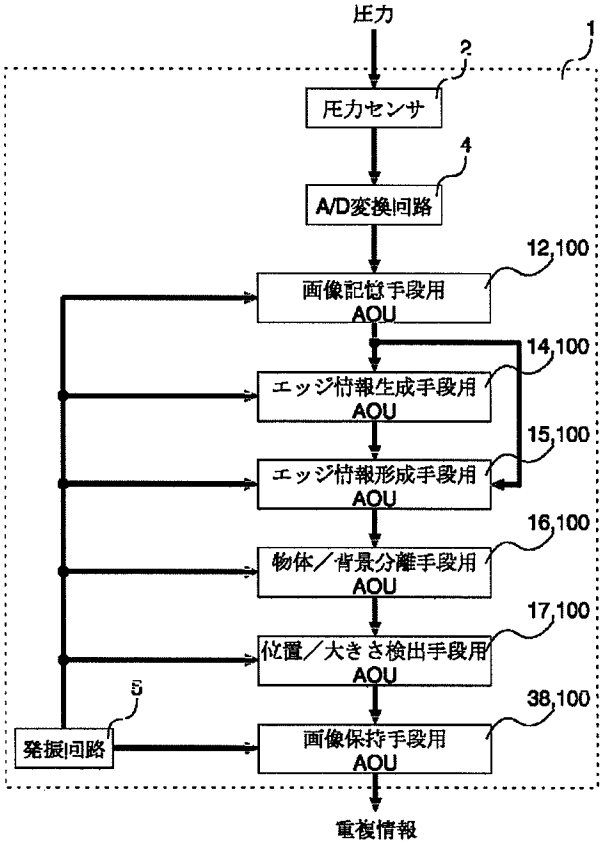
【図2】



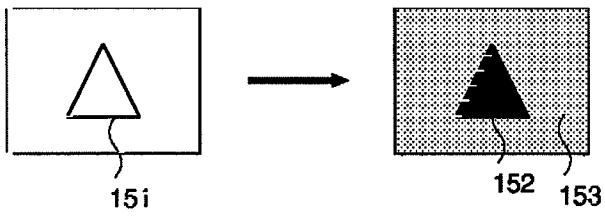
【図3】



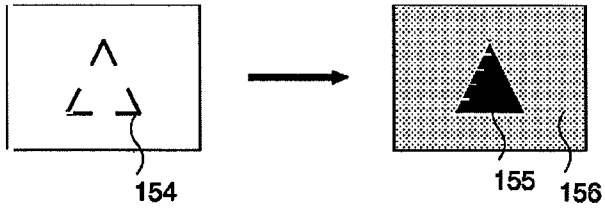
【図4】



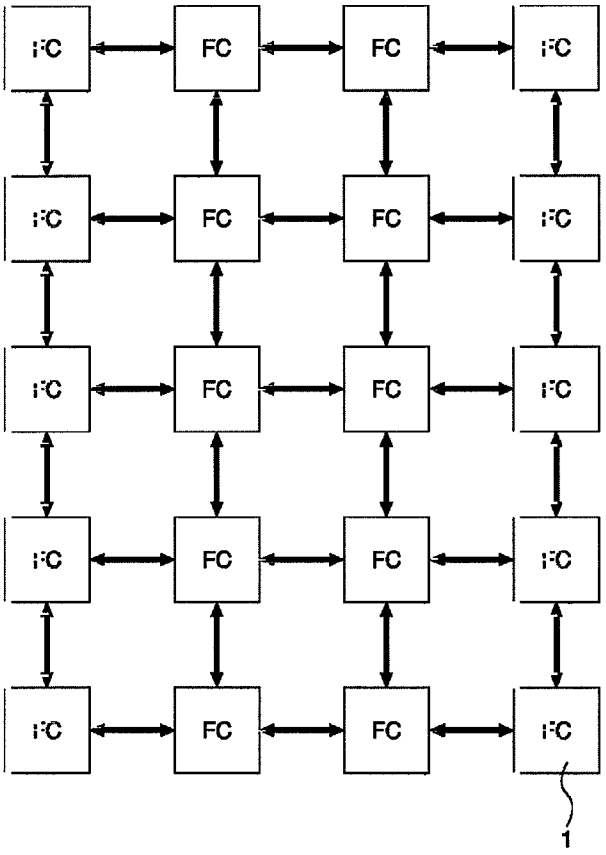
【図21】



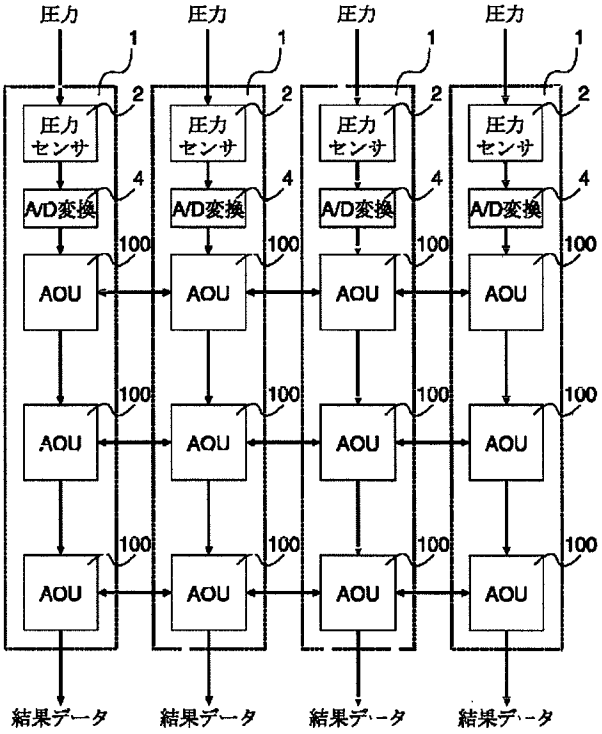
【図23】



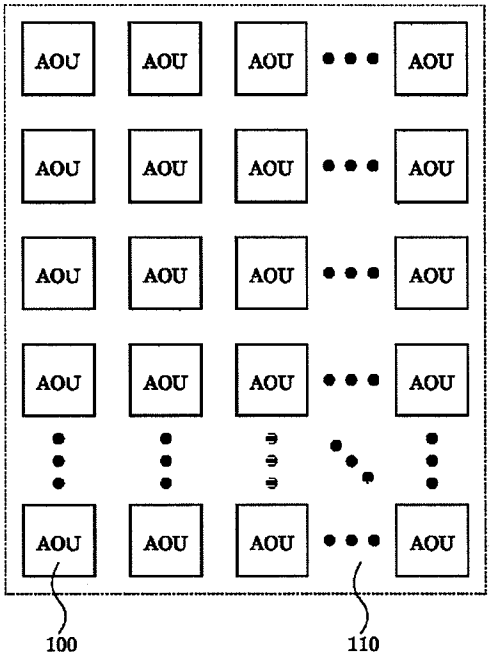
【図5】


FC
1

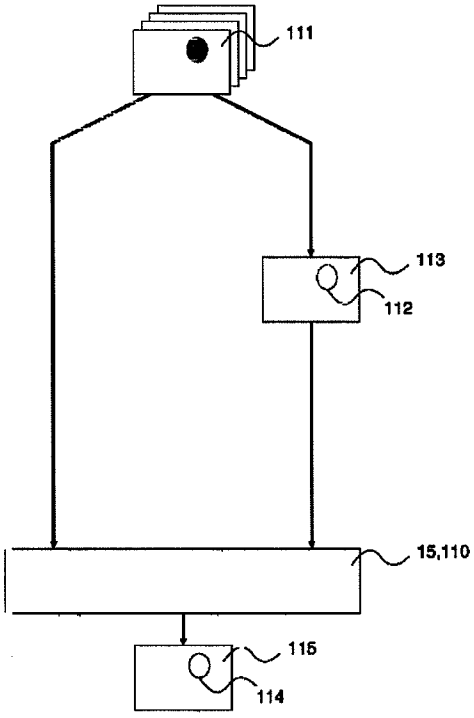
【図6】


圧力
1
2
圧力センサ
4
A/D変換
100
AOU
結果データ

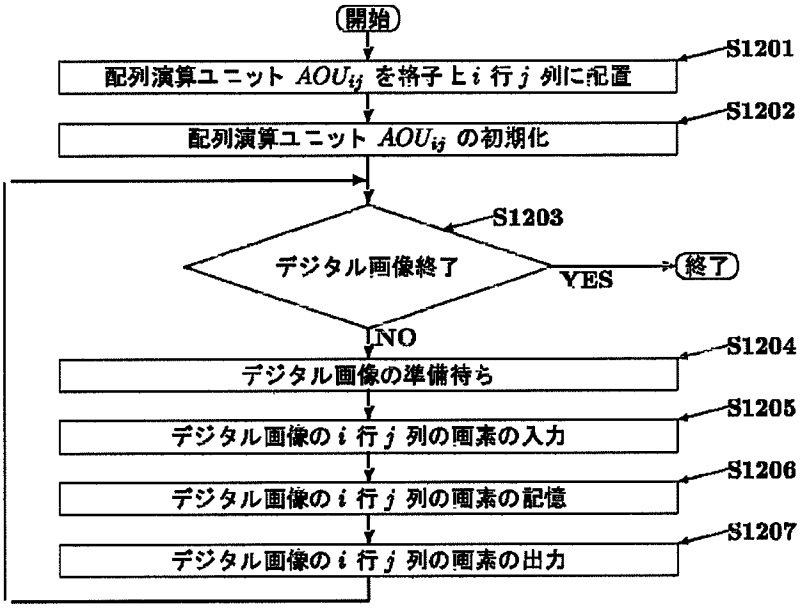
【図7】


AOU
100
110

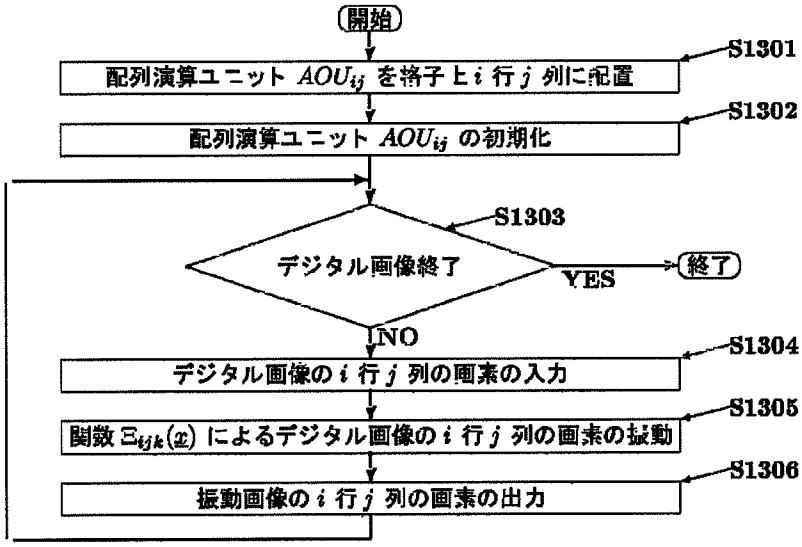
【図11】


111
113
112
15,110
115
114

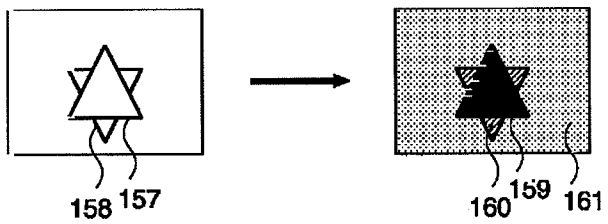
【図8】


開始
配列演算ユニット AOU_{ij} を格子上 i 行 j 列に配置
S1201
配列演算ユニット AOU_{ij} の初期化
S1202
デジタル画像終了
S1203
YES
終了
NO
デジタル画像の準備待ち
S1204
デジタル画像の i 行 j 列の画素の入力
S1205
デジタル画像の i 行 j 列の画素の記憶
S1206
デジタル画像の i 行 j 列の画素の出力
S1207

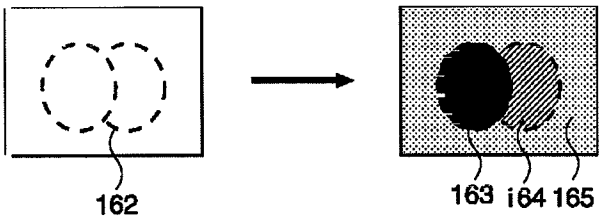
【図9】


開始
配列演算ユニット AOU_{ij} を格子上 i 行 j 列に配置
S1301
配列演算ユニット AOU_{ij} の初期化
S1302
デジタル画像終了
S1303
YES
終了
NO
デジタル画像の i 行 j 列の画素の入力
S1304
関数 $\Xi_{ijk}(x)$ によるデジタル画像の i 行 j 列の画素の振動
S1305
振動画像の i 行 j 列の画素の出力
S1306

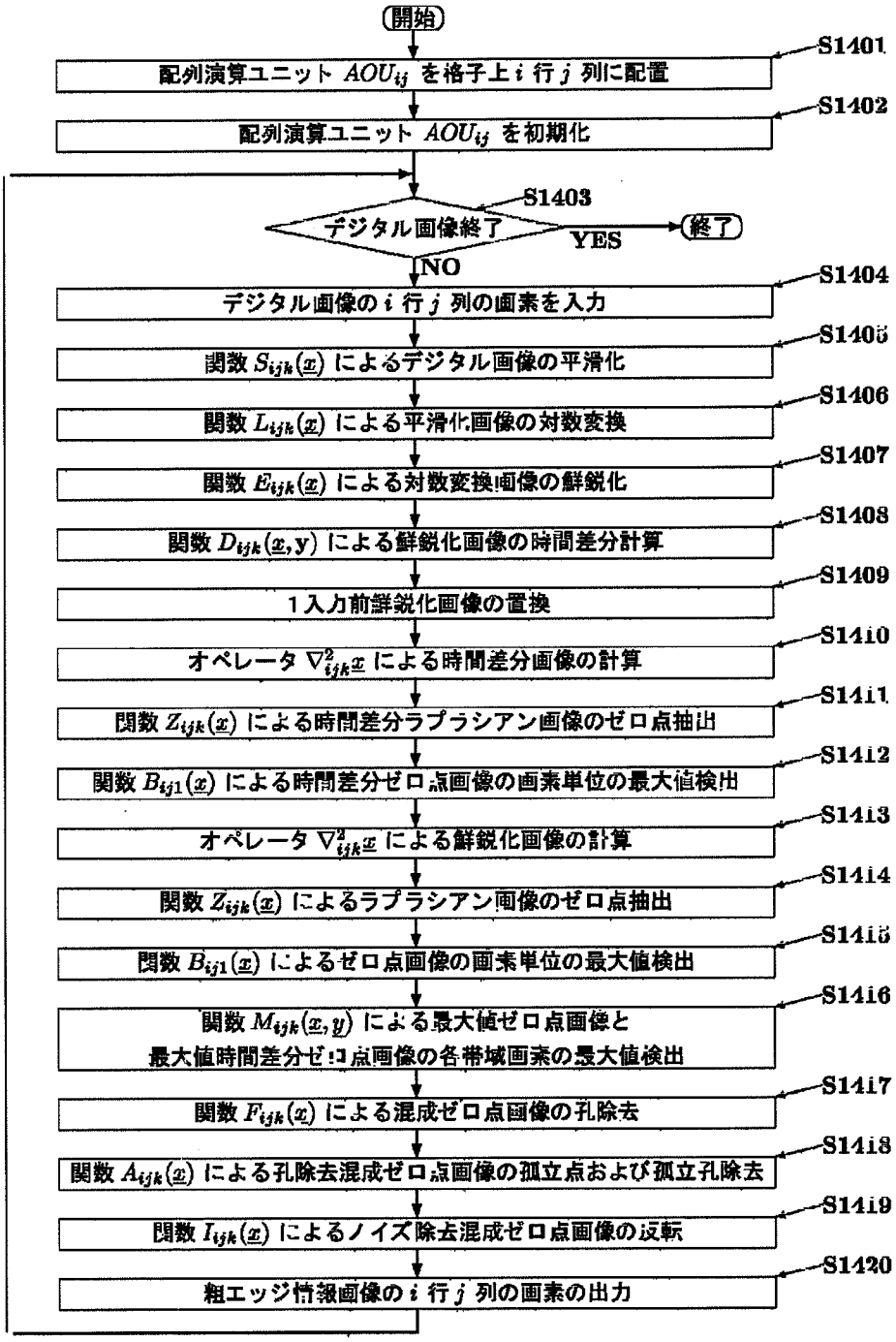
【図24】


158
157
160
159
161

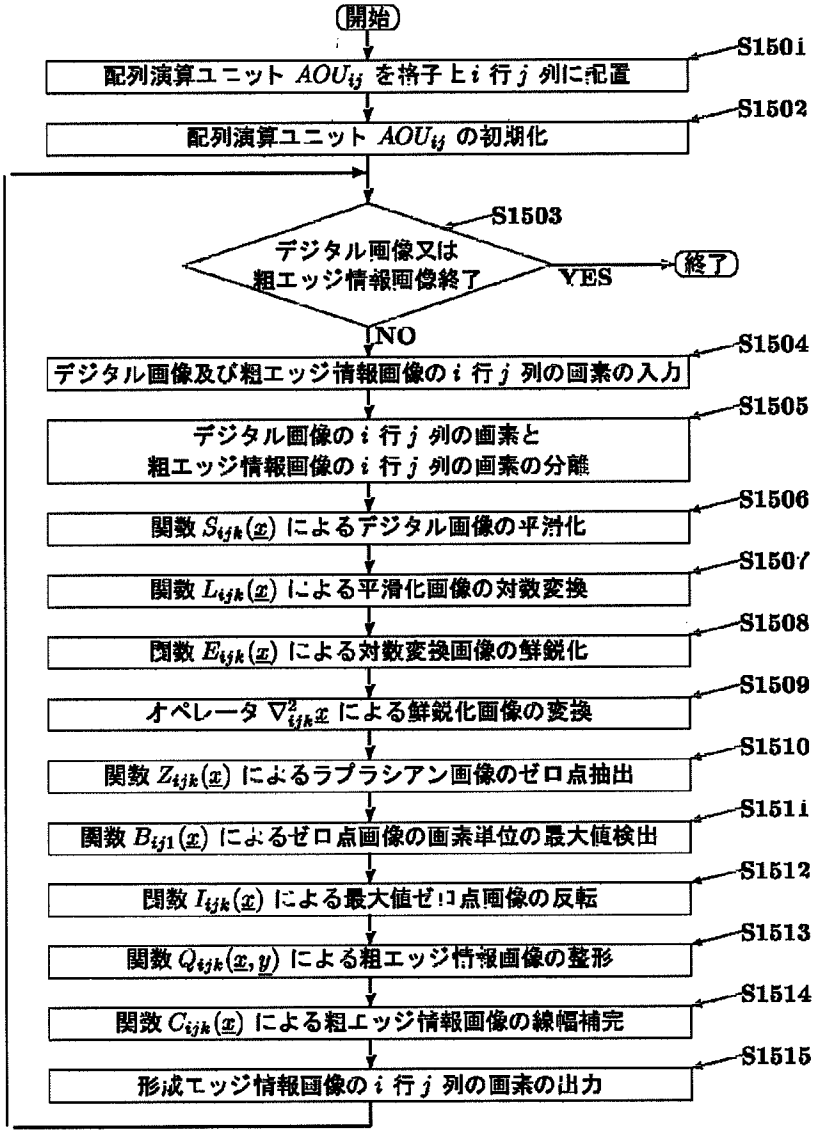
【図25】


162
163
164
165

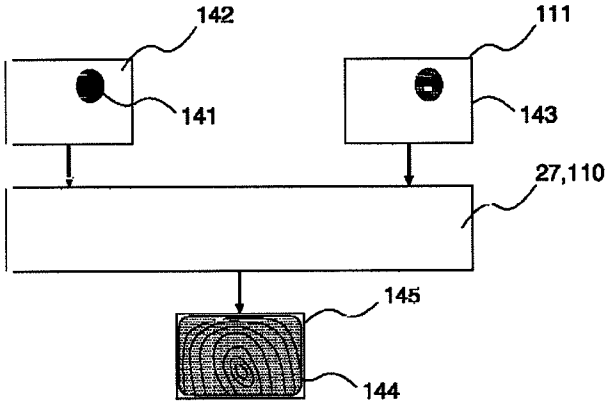
【図10】


開始
S1401
配列演算ユニット AOU_{ij} を格子上 i 行 j 列に配置
S1402
配列演算ユニット AOU_{ij} を初期化
S1403
デジタル画像終了
YES
終了
NO
S1404
デジタル画像の i 行 j 列の画素を入力
S1405
関数 $S_{ijk}(x)$ によるデジタル画像の平滑化
S1406
関数 $L_{ijk}(x)$ による平滑化画像の対数変換
S1407
関数 $E_{ijk}(x)$ による対数変換画像の鮮鋭化
S1408
関数 $D_{ijk}(x,y)$ による鮮鋭化画像の時間差分計算
S1409
1入力前鮮鋭化画像の置換
S1410
オペレータ $\nabla^2_{ijk}x$ による時間差分画像の計算
S1411
関数 $Z_{ijk}(x)$ による時間差分ラプラシアン画像のゼロ点抽出
S1412
関数 $B_{ij1}(x)$ による時間差分ゼロ点画像の画素単位の最大値検出
S1413
オペレータ $\nabla^2_{ijk}x$ による鮮鋭化画像の計算
S1414
関数 $Z_{ijk}(x)$ によるラプラシアン画像のゼロ点抽出
S1415
関数 $B_{ij1}(x)$ によるゼロ点画像の画素単位の最大値検出
S1416
関数 $M_{ijk}(x,y)$ による最大値ゼロ点画像と
最大値時間差分ゼロ点画像の各帯域画素の最大値検出
S1417
関数 $F_{ijk}(x)$ による混成ゼロ点画像の孔除去
S1418
関数 $A_{ijk}(x)$ による孔除去混成ゼロ点画像の孤立点および孤立孔除去
S1419
関数 $I_{ijk}(x)$ によるノイズ除去混成ゼロ点画像の反転
S1420
粗エッジ情報画像の i 行 j 列の画素の出力

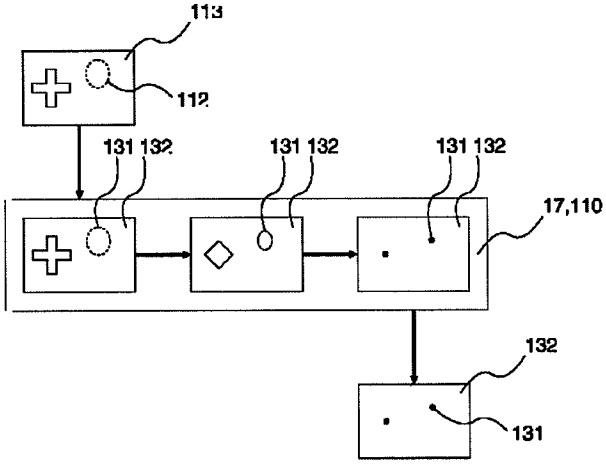
【図12】


開始
配列演算ユニット AOU_{ij} を格子上 i 行 j 列に配置
S1501
配列演算ユニット AOU_{ij} の初期化
S1502
デジタル画像又は粗エッジ情報画像終了
S1503
YES
終了
NO
デジタル画像及び粗エッジ情報画像の i 行 j 列の画素の入力
S1504
デジタル画像の i 行 j 列の画素と粗エッジ情報画像の i 行 j 列の画素の分離
S1505
関数 $S_{ijk}(\underline{x})$ によるデジタル画像の平滑化
S1506
関数 $L_{ijk}(\underline{x})$ による平滑化画像の対数変換
S1507
関数 $E_{ijk}(\underline{x})$ による対数変換画像の鮮鋭化
S1508
オペレータ $\nabla^2_{ijk}\underline{x}$ による鮮鋭化画像の変換
S1509
関数 $Z_{ijk}(\underline{x})$ によるラプラシアン画像のゼロ点抽出
S1510
関数 $B_{ij1}(\underline{x})$ によるゼロ点画像の画素単位の最大値検出
S1511
関数 $I_{ijk}(\underline{x})$ による最大値ゼロ点画像の反転
S1512
関数 $Q_{ijk}(\underline{x},\underline{y})$ による粗エッジ情報画像の整形
S1513
関数 $C_{ijk}(\underline{x})$ による粗エッジ情報画像の線幅補完
S1514
形成エッジ情報画像の i 行 j 列の画素の出力
S1515

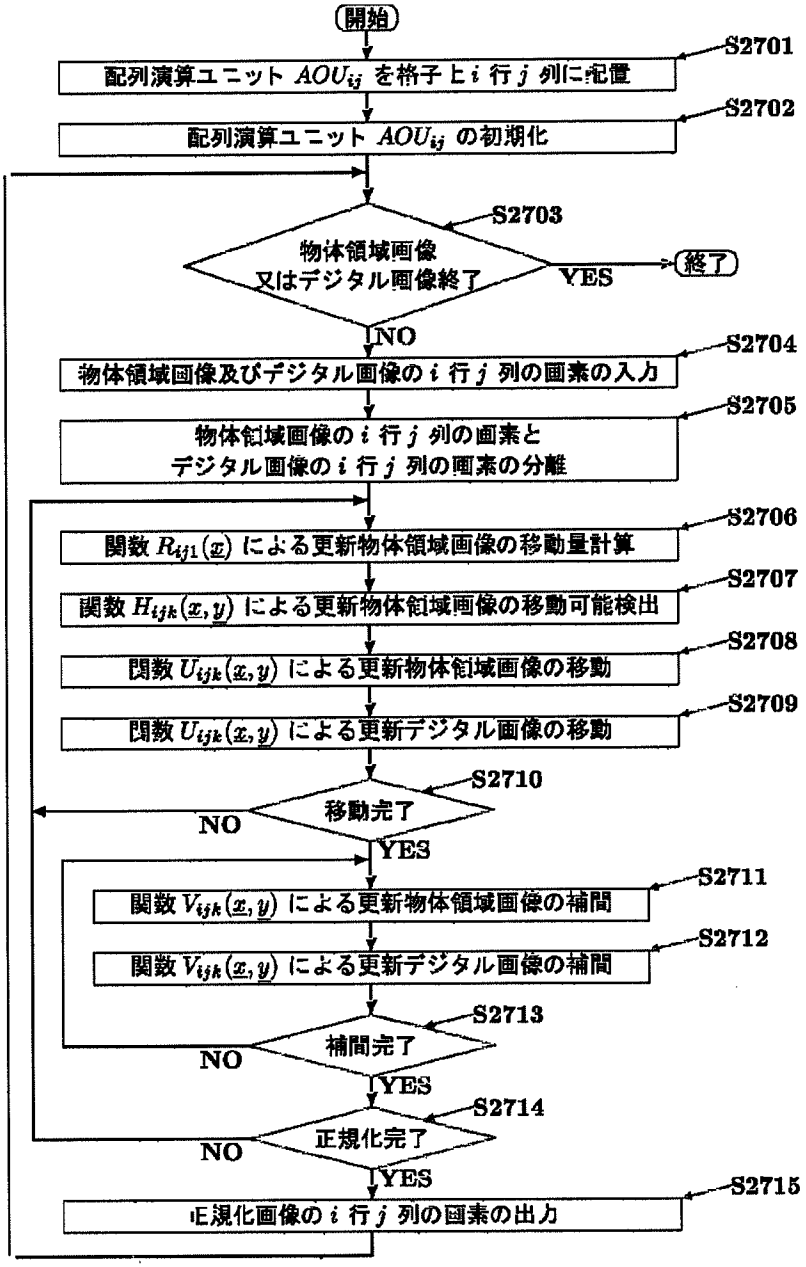
【図13】


142
141
111
143
27,110
145
144

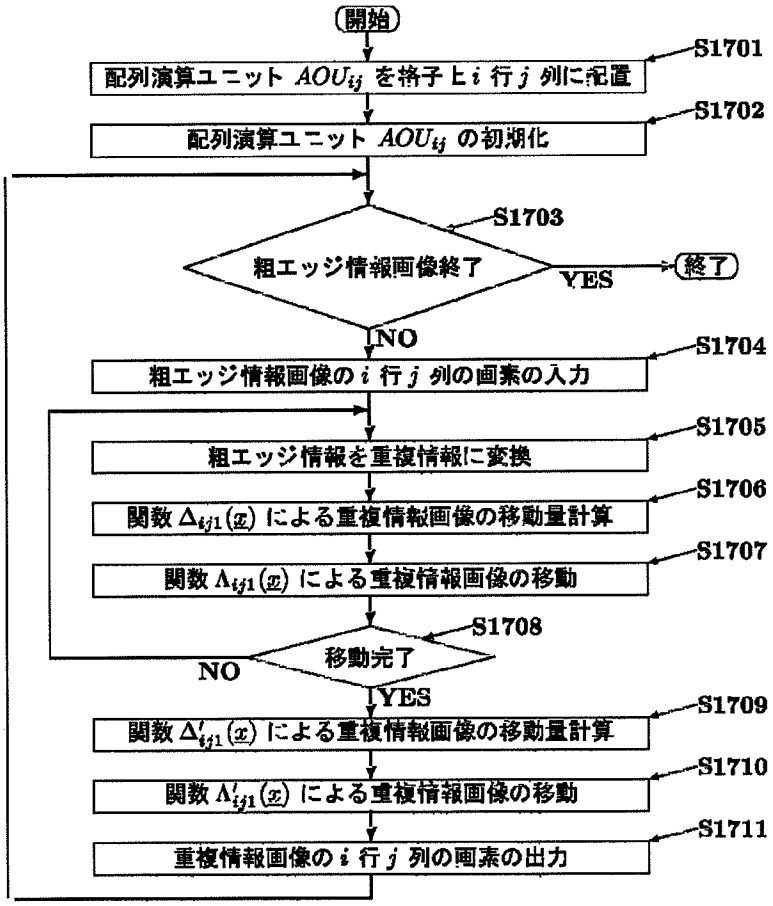
【図15】


113
112
131 132
131 132
131 132
17,110
132
131

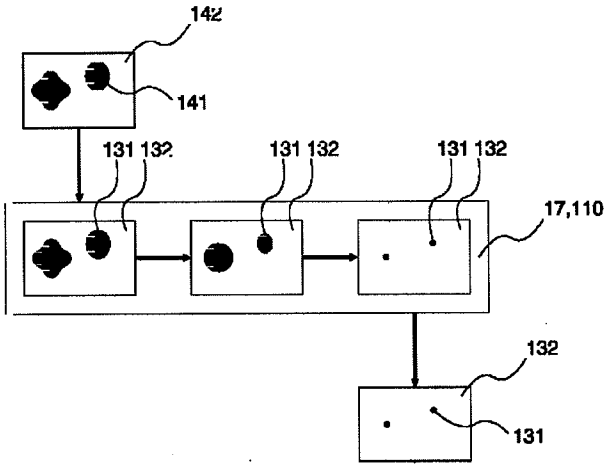
【図14】


開始
S2701
配列演算ユニット AOU_{ij} を格子上 i 行 j 列に配置
S2702
配列演算ユニット AOU_{ij} の初期化
S2703
物体領域画像
又はデジタル画像終了
YES
終了
NO
S2704
物体領域画像及びデジタル画像の i 行 j 列の画素の入力
S2705
物体領域画像の i 行 j 列の画素と
デジタル画像の i 行 j 列の画素の分離
S2706
関数 $R_{ij1}(x)$ による更新物体領域画像の移動量計算
S2707
関数 $H_{ijk}(x,y)$ による更新物体領域画像の移動可能検出
S2708
関数 $U_{ijk}(x,y)$ による更新物体領域画像の移動
S2709
関数 $U_{ijk}(x,y)$ による更新デジタル画像の移動
S2710
移動完了
NO
YES
S2711
関数 $V_{ijk}(x,y)$ による更新物体領域画像の補間
S2712
関数 $V_{ijk}(x,y)$ による更新デジタル画像の補間
S2713
補間完了
NO
YES
S2714
正規化完了
NO
YES
S2715
正規化画像の i 行 j 列の画素の出力

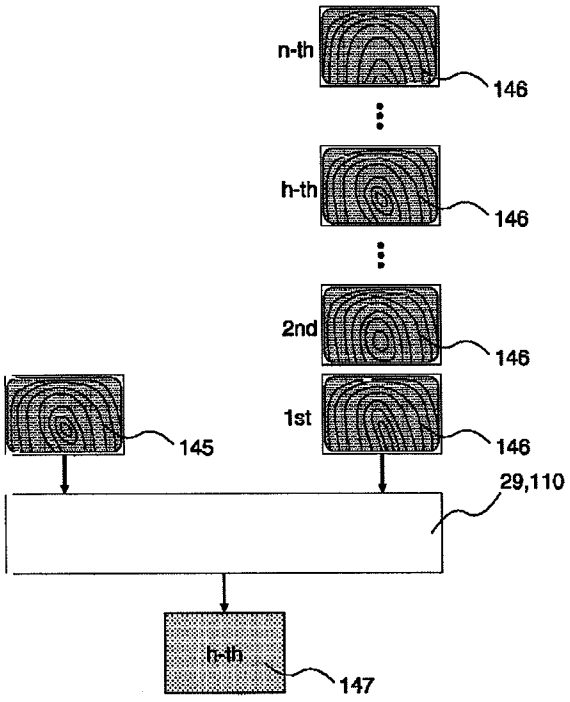
【図16】


開始
S1701
配列演算ユニット AOU_{ij} を格子上 i 行 j 列に配置
S1702
配列演算ユニット AOU_{ij} の初期化
S1703
粗エッジ情報画像終了
YES
終了
NO
S1704
粗エッジ情報画像の i 行 j 列の画素の入力
S1705
粗エッジ情報を重複情報に変換
S1706
関数 $\Delta_{ij1}(x)$ による重複情報画像の移動量計算
S1707
関数 $\Lambda_{ij1}(x)$ による重複情報画像の移動
S1708
移動完了
NO
YES
S1709
関数 $\Delta'_{ij1}(x)$ による重複情報画像の移動量計算
S1710
関数 $\Lambda'_{ij1}(x)$ による重複情報画像の移動
S1711
重複情報画像の i 行 j 列の画素の出力

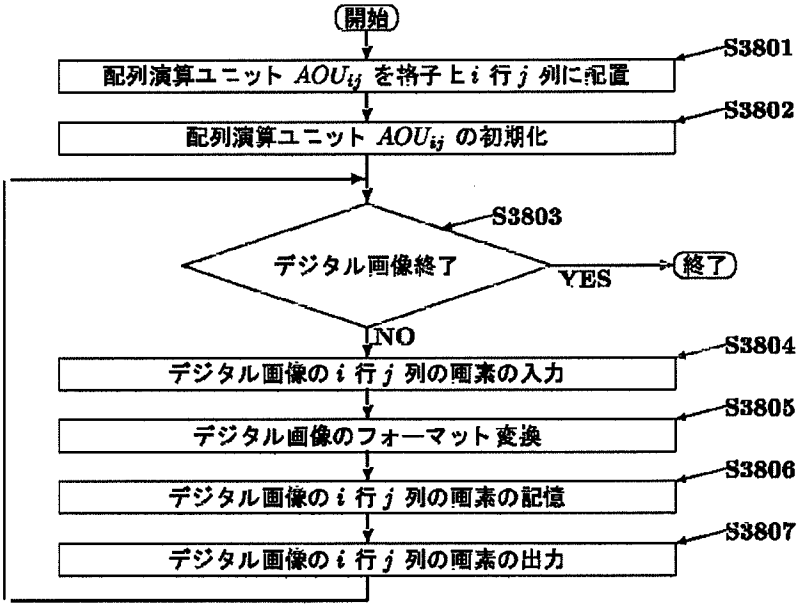
【図17】


142
141
131 132
131 132
131 132
17,110
132
131

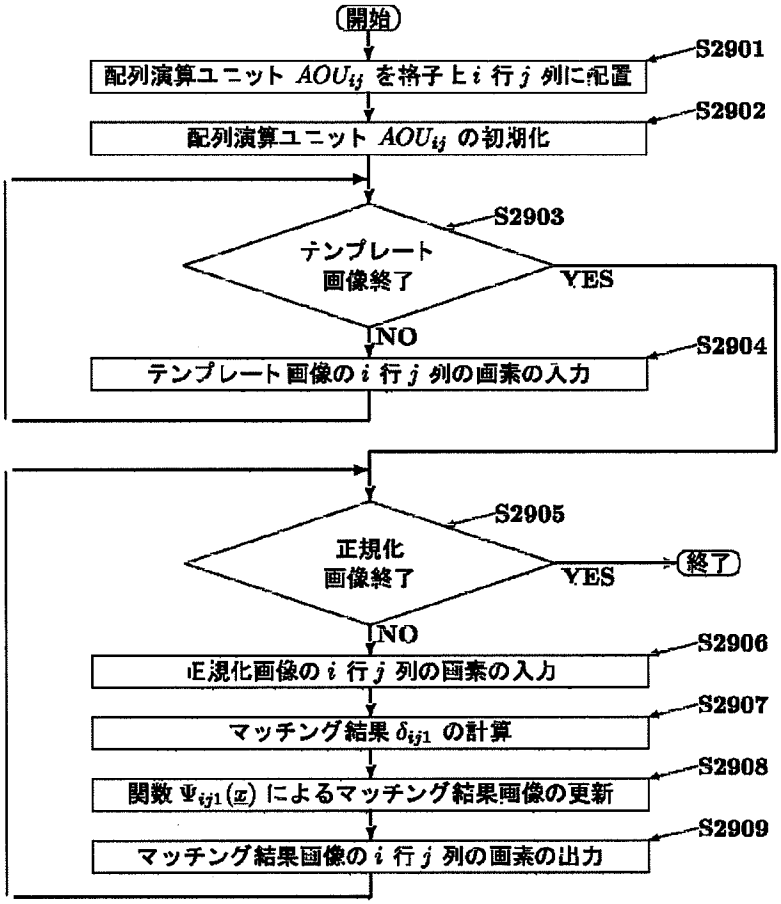
【図19】


n-th
146
h-th
146
2nd
146
1st
146
145
29,110
h-th
147

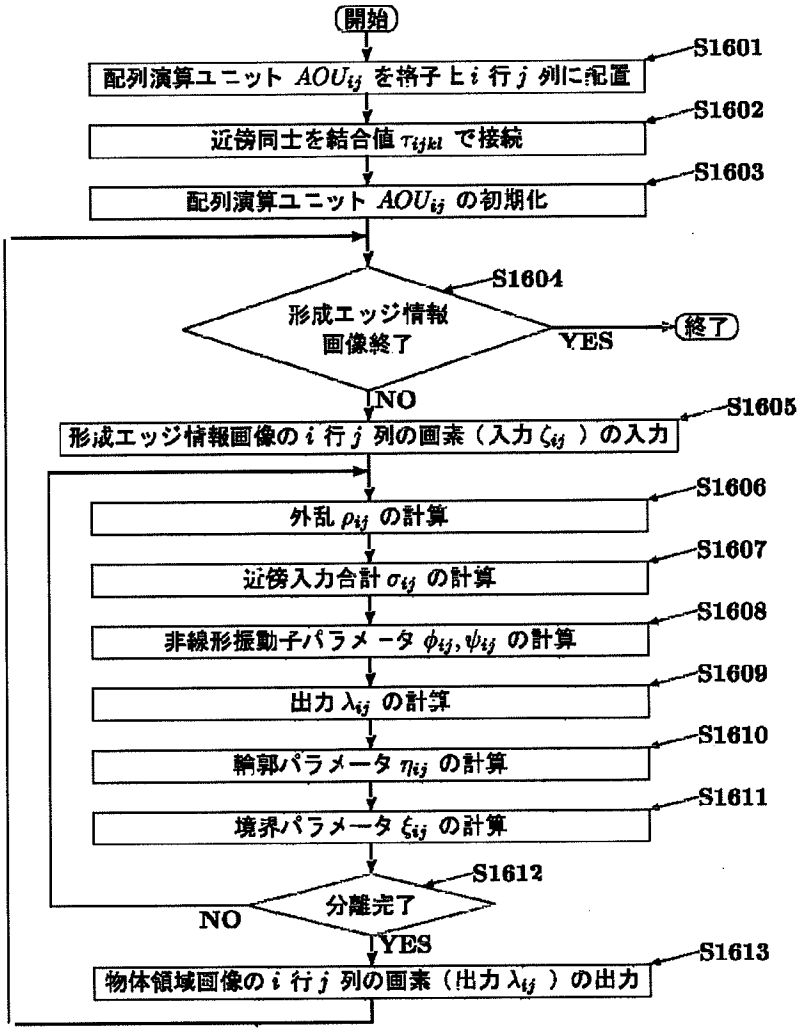
【図18】


開始
S3801
配列演算ユニット AOU_{ij} を格子上 i 行 j 列に配置
S3802
配列演算ユニット AOU_{ij} の初期化
S3803
デジタル画像終了
YES
終了
NO
S3804
デジタル画像の i 行 j 列の画素の入力
S3805
デジタル画像のフォーマット変換
S3806
デジタル画像の i 行 j 列の画素の記憶
S3807
デジタル画像の i 行 j 列の画素の出力

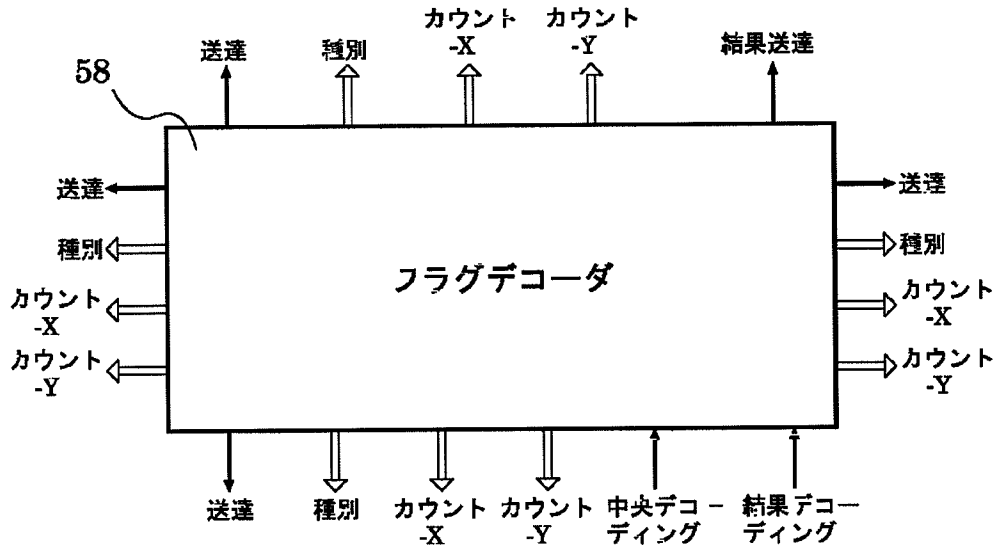
【図20】


開始
S2901
配列演算ユニット AOU_{ij} を格子上 i 行 j 列に配置
S2902
配列演算ユニット AOU_{ij} の初期化
S2903
テンプレート
画像終了
YES
NO
S2904
テンプレート画像の i 行 j 列の画素の入力
S2905
正規化
画像終了
YES
終了
NO
S2906
正規化画像の i 行 j 列の画素の入力
S2907
マッチング結果 δ_{ij1} の計算
S2908
関数 $\Psi_{ij1}(x)$ によるマッチング結果画像の更新
S2909
マッチング結果画像の i 行 j 列の画素の出力

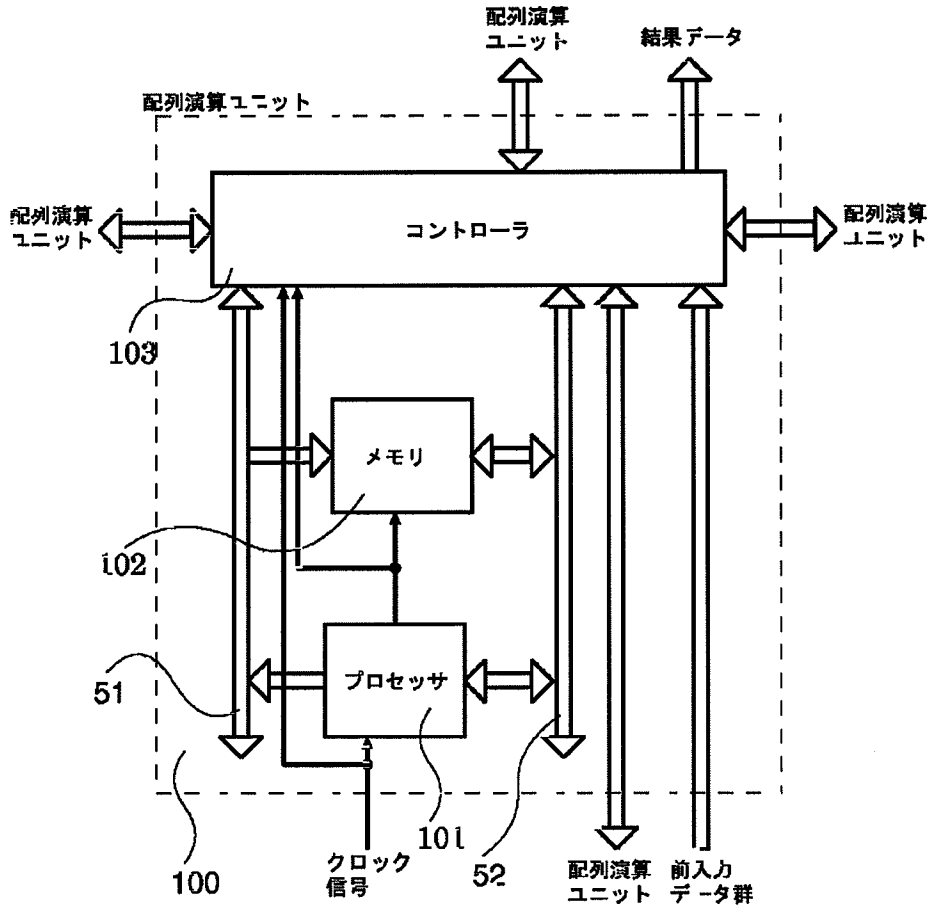
【図22】


開始
S1601
配列演算ユニット AOU_{ij} を格子上 i 行 j 列に配置
S1602
近傍同士を結合値 τ_{ijkl} で接続
S1603
配列演算ユニット AOU_{ij} の初期化
S1604
形成エッジ情報画像終了
YES
終了
NO
S1605
形成エッジ情報画像の i 行 j 列の画素（入力 ζ_{ij} ）の入力
S1606
外乱 ρ_{ij} の計算
S1607
近傍入力合計 σ_{ij} の計算
S1608
非線形振動子パラメータ ϕ_{ij}, ψ_{ij} の計算
S1609
出力 λ_{ij} の計算
S1610
輪郭パラメータ η_{ij} の計算
S1611
境界パラメータ ξ_{ij} の計算
S1612
分離完了
NO
YES
S1613
物体領域画像の i 行 j 列の画素（出力 λ_{ij} ）の出力

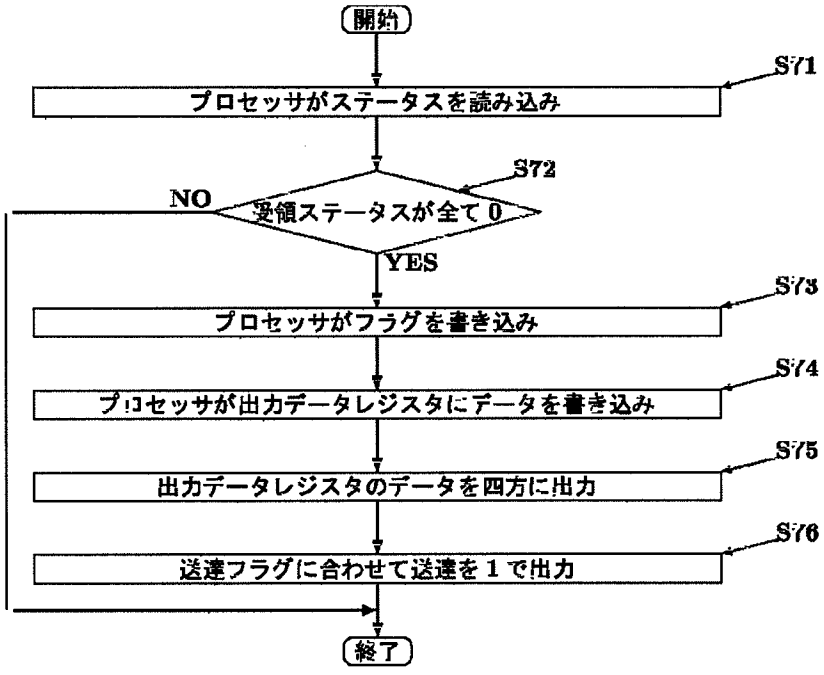
【図28】


58
送達
種別
カウント -X
カウント -Y
結果送達
フラグデコーダ
送達
種別
カウント -X
カウント -Y
送達
種別
カウント -X
カウント -Y
中央デコーディング
結果デコーディング

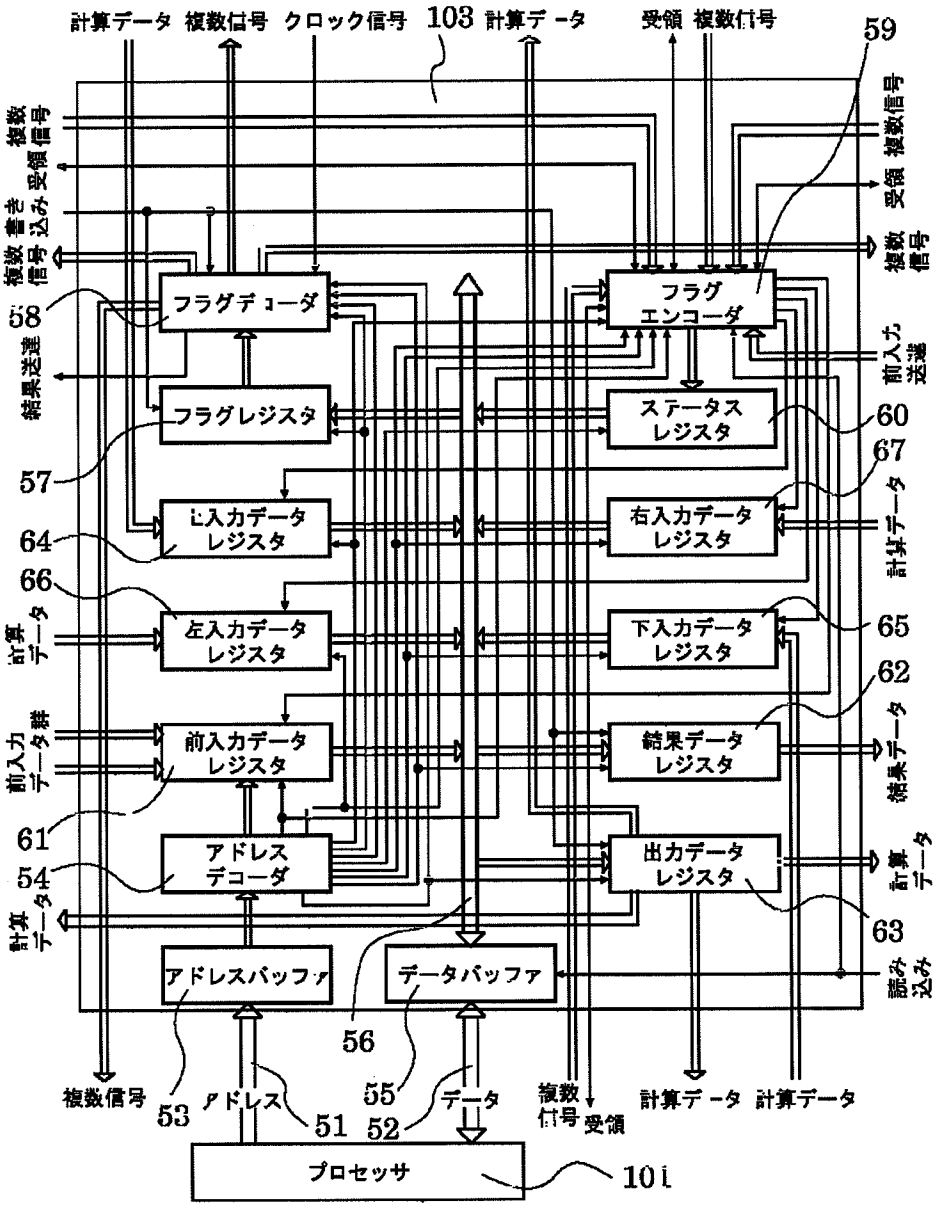
【図26】


配列演算ユニット
結果データ
配列演算ユニット
配列演算ユニット
コントローラ
配列演算ユニット
103
メモリ
102
プロセッサ
51
100
クロック信号
101
52
配列演算ユニット
前入力データ群

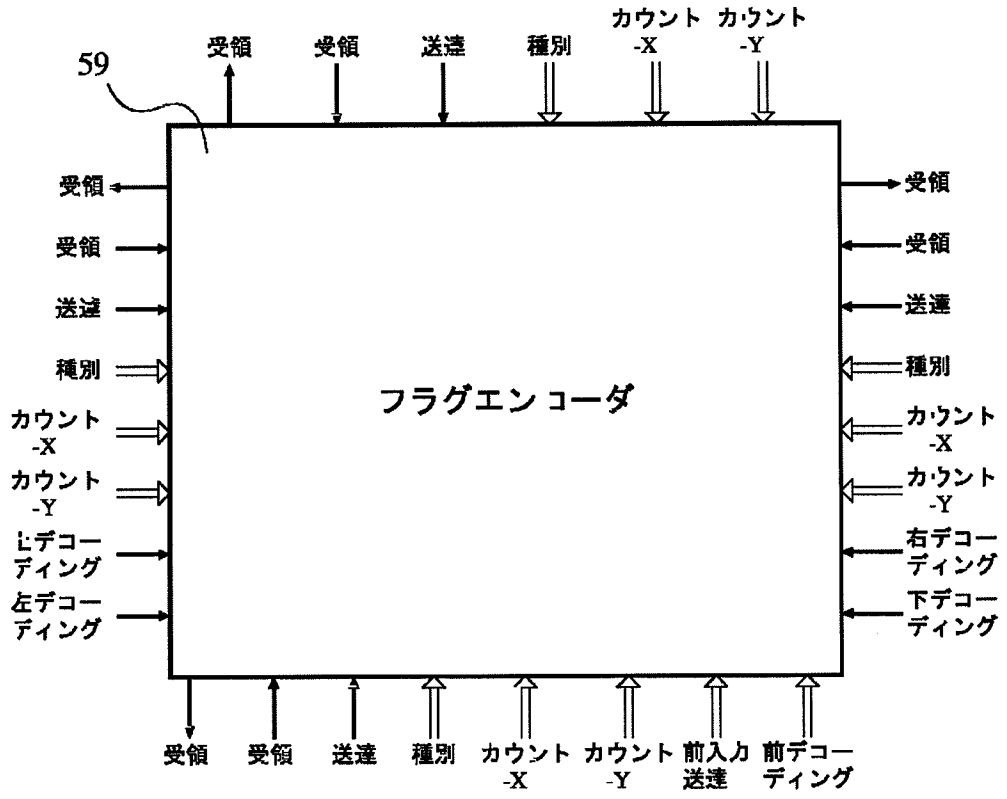
【図30】


開始
S71
プロセッサがステータスを読み込み
S72
NO
受領ステータスが全て0
YES
S73
プロセッサがフラグを書き込み
S74
プロセッサが出力データレジスタにデータを書き込み
S75
出力データレジスタのデータを四方に出力
S76
送達フラグに合わせて送達を1で出力
終了

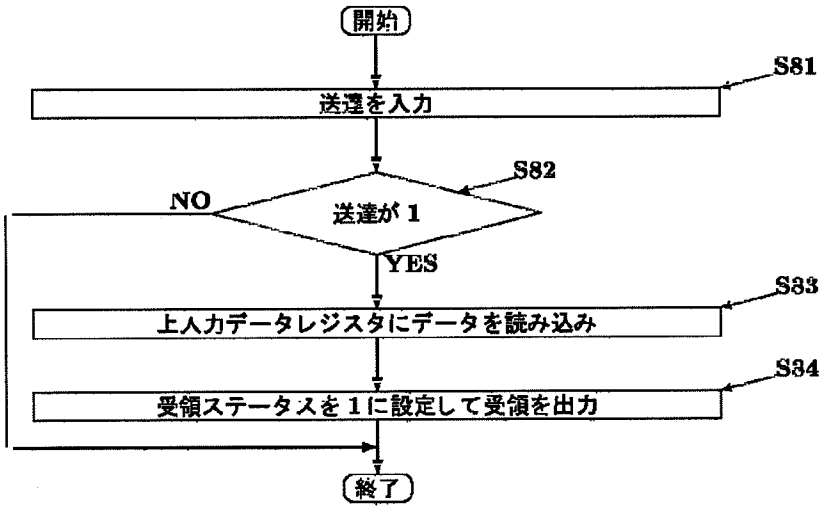
【図27】


計算データ
複数信号
クロック信号
103
計算データ
受領
複数信号
59
複数信号
受領
複数書き込み信号
複数信号
58
フラグデコーダ
フラグエンコーダ
結果送達
前入力送達
フラグレジスタ
ステータスレジスタ
60
57
67
上入力データレジスタ
右入力データレジスタ
計算データ
64
66
左入力データレジスタ
下入力データレジスタ
65
計算データ
62
前入力データ群
前入力データレジスタ
結果データレジスタ
結果データ
61
アドレスデコーダ
出力データレジスタ
54
計算データ
63
計算データ
アドレスバッファ
データバッファ
読み込み
56
複数信号
53
アドレス
51
55
データ
52
複数信号
受領
計算データ
計算データ
プロセッサ
101

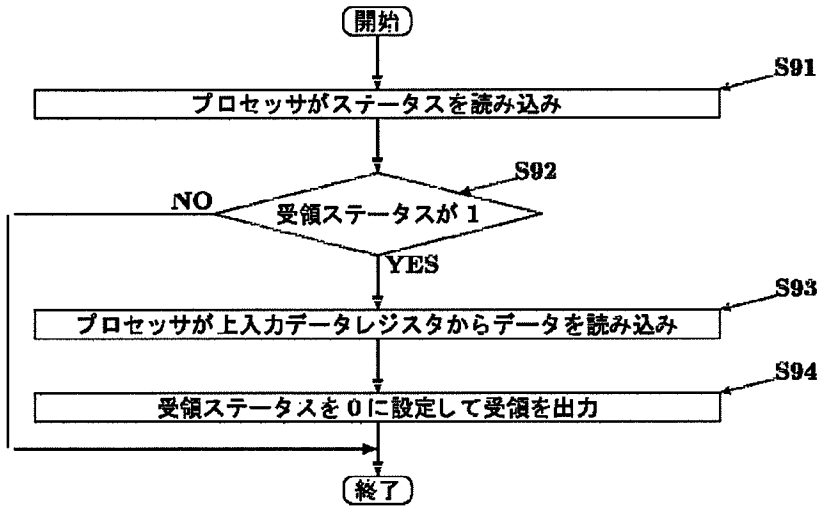
【図29】


59
受領
受領
送達
種別
カウント -X
カウント -Y
受領
受領
送達
種別
カウント -X
カウント -Y
上デコーディング
左デコーディング
フラグエンコーダ
受領
受領
送達
種別
カウント -X
カウント -Y
前入力送達
前デコーディング
受領
受領
送達
種別
カウント -X
カウント -Y
右デコーディング
下デコーディング

【図31】


開始
S81
送達を入力
S82
NO
送達が 1
YES
S83
上人力データレジスタにデータを読み込み
S84
受領ステータスを 1 に設定して受領を出力
終了

【図32】



フロントページの続き

Fターム(参考) 2F051 AA21 AB03 AB04 AB06 AC01
AC09 BA07
2F063 AA03 AA41 AA48 BA29 BA30
BB01 BD01 BD05 CB00 DA02
DD07 EB02 EC00 KA01 LA12
LA18 LA19 LA23 LA26 LA30
NA02 NA06
2F069 AA03 AA13 AA64 AA96 GG02
GG06 GG16 GG20 GG58 GG59
HH30 NN00 NN08 NN25